

Lecture Notes in Artificial Intelligence 1394

Subseries of Lecture Notes in Computer Science

Xindong Wu Ramamohanarao Kotagiri
Kevin B. Korb (Eds.)

Research and Development in Knowledge Discovery and Data Mining

Second Pacific-Asia Conference, PAKDD-98
Melbourne, Australia, April 1998
Proceedings



Springer

Lecture Notes in Artificial Intelligence 1394

Subseries of Lecture Notes in Computer Science

Edited by J. G. Carbonell and J. Siekmann

Lecture Notes in Computer Science

Edited by G. Goos, J. Hartmanis and J. van Leeuwen

Xindong Wu Ramamohanarao Kotagiri
Kevin B. Korb (Eds.)

Research and Development in Knowledge Discovery and Data Mining

Second Pacific-Asia Conference, PAKDD-98
Melbourne, Australia, April 15-17, 1998
Proceedings



Springer

Series Editors

Jaime G. Carbonell, Carnegie Mellon University, Pittsburgh, PA, USA

Jörg Siekmann, University of Saarland, Saarbrücken, Germany

Volume Editors

Xindong Wu

School of Computer Science and Software Engineering, Monash University

900 Dandenong Road, Caulfield East, Victoria 3145, Australia

E-mail: Xindong.Wu@fcit.monash.edu.au

Ramamohanrao Kotagiri

Department of Computer Science, The University of Melbourne

Parkville, Victoria 3052, Australia

E-mail: rao@cs.mu.OZ.AU

Kevin B. Korb

School of Computer Science and Engineering, Monash University

Clayton, Victoria 3168, Australia

E-mail: Kevin.Korb@fcit.monash.edu.au

Cataloging-in-Publication Data applied for

Die Deutsche Bibliothek - CIP-Einheitsaufnahme

Research and development in knowledge discovery and data mining : second Asia Pacific conference ; proceedings / PAKDD-98, Melbourne, Australia, April 15 - 17, 1998. Xindong Wu ... (ed.). - Berlin ; Heidelberg ; New York ; Barcelona ; Budapest ; Hong Kong ; London ; Milan ; Paris ; Santa Clara ; Singapore ; Tokyo : Springer, 1998

(Lecture notes in computer science ; Vol. 1394 : Lecture notes in artificial intelligence)

ISBN 3-540-64383-4

CR Subject Classification (1991): I.2, H.3, H.5.1-2, G.3, J.1

ISSN 0302-9743

ISBN 3-540-64383-4 Springer-Verlag Berlin Heidelberg New York

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer -Verlag. Violations are liable for prosecution under the German Copyright Law.

© Springer-Verlag Berlin Heidelberg 1998

Printed in Germany

Typesetting: Camera ready by author

SPIN 10635950 06/3142 - 5 4 3 2 1 0 Printed on acid-free paper

Preface

The Second Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD-98) provides an international forum for the sharing of original research results and practical development experiences among researchers and practitioners from different KDD related areas such as machine learning, databases, statistics, knowledge acquisition, data visualization, software re-engineering, and knowledge-based systems. It follows the success of PAKDD-97 held in Singapore in 1997, bringing together participants from universities, industry, and government.

PAKDD-98 covers broad and diverse topics in knowledge discovery and data mining, including:

- Data and dimensionality reduction
- Data mining algorithms and tools
- Data mining and data warehousing
- Data mining on the Internet
- Data mining metrics
- Data preprocessing and postprocessing
- Data and knowledge visualization
- Deduction and induction in KDD
- Discretization of continuous data
- Distributed data mining
- KDD framework and process
- Knowledge representation and acquisition in KDD
- Knowledge reuse and role of domain knowledge
- Knowledge acquisition in software re-engineering and software Information Systems
- Induction of rules and decision trees
- Management issues in KDD
- Machine learning, statistical and visualization aspects of KDD (including neural networks, rough set theory, and inductive logic programming)
- Mining in-the-large versus mining in-the-small
- Noise handling
- Security and privacy issues in KDD
- Successful/Innovative KDD applications in science, government, business, and industry

Of the 110 submissions, we accepted 31 regular papers;¹ an acceptance rate of 28%. In addition, over 20 papers were accepted as posters for short presentations at the conference.

The conference program could not have been achieved without the contributions of many individuals. First, we would like to thank the invited speakers — Jiawei Han (ACSys Keynote Speaker), Chris Wallace and Bhavani Thuraisingham — and the Conference Chairs — Ross Quinlan and Bala Srinivasan —

¹ One was later withdrawn.

whose involvement and support have added greatly to the quality of the conference. Our sincere gratitude goes to all of the authors who submitted papers and to all program committee members and reviewers who provided high quality reviews. Our special appreciation goes to Lipo Wang, the Publicity Chair, who has spent countless hours on advertising the conference in various mailing lists, news groups, newspapers and magazines, and Michelle Riseley, the Conference Treasurer, who is always well-organized and very efficient.

PAKDD-98 proudly acknowledges major sponsorships from the Monash Artificial Intelligence Consortium (MAIC), the School of Computer Science and Software Engineering at Monash University, and ACSys (The Cooperative Research Center for Advanced Computational Systems). The conference is also supported by the Department of Computer Science at Melbourne University, the Faculty of Engineering at Deakin University, and Yamaguchi University.

February 1998

Xindong Wu
Ramamohanarao Kotagiri
Kevin B. Korb

PAKDD-98 Conference Committee

Conference Chairs:

J. Ross Quinlan Bala Srinivasan	RuleQuest Research Pty Ltd Monash University
------------------------------------	---

Program Chairs:

Xindong Wu Ramamohanarao Kotagiri	Monash University Melbourne University
--------------------------------------	---

Organizing Committee Co-Chairs:

Kevin B. Korb Graham Williams	Monash University CSIRO
----------------------------------	----------------------------

PAKDD-98 Publicity Chair:

Lipo Wang	Deakin University
-----------	-------------------

PAKDD-98 Tutorial Chair:

Jon Oliver	Monash University
------------	-------------------

PAKDD-98 Treasurer:

Michelle Riseley	Monash University
------------------	-------------------

Program Committee

Grigoris Antoniou	Griffith University
James Boyce	Kings College, London
Ivan Bratko	Ljubljana University
Mike Cameron-Jones	University of Tasmania
Arbee Chen	National Tsing Hua University, Taiwan
David Cheung	Hong Kong University
Vic Ciesielski	Royal Melbourne Institute of Technology
Honghua Dai	Monash University
John Debenham	University of Technology, Sydney
Olivier de Vel	James Cook University
Tharam Dillon	La Trobe University
Guozhu Dong	University of Melbourne
Peter Eklund	University of Adelaide
Usama Fayyad	Microsoft Research, USA
Matjaz Gams	Jozef Stefan Institute
Yike Guo	Imperial College, London
Jiawei Han	Simon Fraser University
David Hand	Open University, UK
Evan Harris	University of Melbourne
David Heckerman	Microsoft Research, USA
David Kemp	University of Melbourne
Masaru Kitsuregawa	Tokyo University
Kevin Korb	Monash University
Hingyan Lee	Japan Singapore AI Center
Jae-Kyu Lee	KAIST, Korea
Deyi Li	Beijing System Engineering Institute
T.Y. Lin	San Jose State University
Bing Liu	National University of Singapore
Huan Liu	National University of Singapore
Zhi-Qiang Liu	University of Melbourne
Hongjun Lu	National University of Singapore
Dickson Lukose	University of New England
Kia Makki	University of Nevada, Las Vegas
Heikki Mannila	University of Helsinki
Peter Milne	CSIRO
Shinichi Morishita	University of Tokyo
Hiroshi Motoda	Osaka University
Hwee-Leng Ong	Japan Singapore AI Center
Jonathan Oliver	Monash University
Maria Orlowska	University of Queensland
Gregory Piatetsky-Shapiro	Knowledge Stream Partners, USA

Niki Pissinou	University of Southwestern Louisiana
Peter Ross	Edinburgh University
Claude Sammut	University of New South Wales
S. Seshadri	IIT Bombay
Hayri Sever	Hacettepe University
Arun Sharma	University of New South Wales
Heinz Schmidt	Monash University
Evangelos Simoudis	IBM, USA
Atsuhiro Takasu	NCSIS, Japan
Takao Terano	University of Tsukuba
Bhavani Thuraisingham	MITRE, USA
Kai Ming Ting	Waikato University
David Urpani	CSIRO
Ramasamy Uthurusamy	General Motors, USA
Lipo Wang	Deakin University
Geoff Webb	Deakin University
Graham Williams	CSIRO
Beat Wuthrich	Hong Kong University of Science & Technology
Xin Yao	ADFA/University of New South Wales
John Zeleznikow	La Trobe University
Diancheng Zhang	Hefei University of Technology
Ming Zhao	Telstra
Zijian Zheng	Deakin University
Ning Zhong	Yamaguchi University
Justin Zobel	Royal Melbourne Institute of Technology

PAKDD-98 Reviewers

Grigoris Antoniou	Kevin Korb
Linda Bird	Rao Kotagiri
Warren F. Bloomer	Hing Yan LEE
Jimmy Boyce	Luke Lake
Ivan Bratko	Deyi Li
Andrew Burrow	Edgar C.H. Lin
Mike Cameron-Jones	T.Y. Lin
C.M. Chang	Bing Liu
Jaturon Chatratchat	Chengfei Liu
Arbee Chen	Huan Liu
Qing Chen	Zhi-Qiang Liu
David Cheung	Hongjun Lu
Bark C. Chiu	Ying Lu
Vincent Cho	Eric McCreath
Vic Ciesielski	Gabor Melli
Richard Cole	Peter Milne
John Crossley	Shinichi Morishita
Honghua Dai	Hiroshi Motoda
Hung Dang	Julian Neil
John Debenham	Kian Sing Ng
Olivier de Vel	Ann Nicholson
Tharam Dillon	Hwee Leng ONG
Guozhu Dong	Jon Oliver
Curtis Dyreson	Maria Orlowska
Peter Eklund	Peter D.L. Pan
Usama Fayyad	Gregory Piatetsky-Shapiro
Matjaz Gams	Niki Pissinou
Mark Grundy	M.V. Ramakrishna
Yike Guo	Peter Ross
Jiawei Han	Claude Sammut
David Hand	Tobias Scheffer
Michael Harries	S. Seshadri
Evan Harris	Sabrina Sestito
Markus Hegland	Arun Sharma
Tadashi Horiuchi	Moninder Singh
Farhad Hussain	Janjao Sutiwaraphun
Micheline Kamber	Takao Terano
David Kemp	Kai Ming Ting
Rakiv Khosla	D. Truffet
Masaru Kitsuregawa	David Urpani
Martin Kolher	Takashi Washio
Krzysztof Koperski	Geoff Webb

Liang Hwang Wen
Ross Wilkinson
Graham Williams
Hugh Williams
Lara Winstone
Y.H. Wu
Beat Wuthrich
Dan Yang
Jun Yao
Xin Yao
Y.Y. Yao
John Zeleznikow
Ming Zhao
Zijian Zheng
Ning Zhong
Justin Zobel

Contents

Papers

Knowledge Acquisition for Goal Prediction in a Multi-user Adventure Game <i>D. W. Albrecht, A. E. Nicholson and I. Zukerman</i>	1
Hybrid Data Mining Systems: The Next Generation <i>S. S. Anand and J. G. Hughes</i>	13
Discovering Case Knowledge Using Data Mining <i>S. S. Anand, D. Patterson, J. G. Hughes and D.A. Bell</i>	25
Discovery of Association Rules over Ordinal Data: A New and Faster Algorithm and Its Application to Basket Analysis <i>O. Büchter and R. Wirth</i>	36
Effect of Data Skewness in Parallel Mining of Association Rules <i>D. W. Cheung and Y. Xiao</i>	48
Trend Directed Learning: A Case Study <i>H. Dai</i>	61
Interestingness of Discovered Association Rules in Terms of Neighborhood-Based Unexpectedness <i>G. Dong and J. Li</i>	72
Point Estimation Using the Kullback-Leibler Loss Function and MML <i>D. L. Dowe, R. A. Baxter, J. J. Oliver and C. S. Wallace</i>	87
Single Factor Analysis in MML Mixture Modelling <i>R. T. Edwards and D. L. Dowe</i>	96
Discovering Associations in Spatial Data – An Efficient Medoid Based Approach <i>V. Estivill-Castro and A. T. Murray</i>	110
Data Mining Using Dynamically Constructed Recurrent Fuzzy Neural Networks <i>Y. Frayman and L. Wang</i>	122
CCAIIA: Clustering Categorical Attributes into Interesting Association Rules <i>B. Gray and M. E. Orlowska</i>	132

Selective Materialization: An Efficient Method for Spatial Data Cube Construction <i>J. Han, N. Stefanovic and K. Koperski</i>	144
Mining Market Basket Data Using Share Measures and Characterized Itemsets <i>R. J. Hilderman, C. L. Carter, H. J. Hamilton and N. Cercone</i>	159
Automatic Visualization Method for Visual Data Mining <i>Y. Iizuka, H. Shiohara, T. Iizuka and S. Isobe</i>	171
Rough Set-Inspired Approach to Knowledge Discovery in Business Databases <i>W. Kowalczyk and Z. Piasta</i>	186
Representative Association Rules <i>M. Kryszkiewicz</i>	198
Identifying Relevant Databases for Multidatabase Mining <i>H. Liu, H. Lu and J. Yao</i>	210
Minimum Message Length Segmentation <i>J. J. Oliver, R. A. Baxter and C.S. Wallace</i>	222
Bayesian Classification Trees with Overlapping Leaves Applied to Credit-Scoring <i>G. Paass and J. Kindermann</i>	234
Contextual Text Representation for Unsupervised Knowledge Discovery in Texts <i>P. Perrin and F. Petry</i>	246
Treatment of Missing Values for Association Rules <i>A. Ragel and B. Crémilleux</i>	258
Mining Regression Rules and Regression Trees <i>B.-Y. Sher, S.-C. Shao and W.-S. Hsieh</i>	271
Mining Algorithms for Sequential Patterns in Parallel: Hash Based Approach <i>T. Shintani and M. Kitsuregawa</i>	283
Wavelet Transform in Similarity Paradigm <i>Z. R. Struzik and A. Siebes</i>	295

Improved Rule Discovery Performance on Uncertainty <i>M. R. Tolun, H. Sever and M. Uludağ</i>	310
Feature Mining and Mapping of Collinear Data <i>O. de Vel, D. Coomans and S. Patrick</i>	322
Knowledge Discovery in Discretionary Legal Domains <i>J. Zeleznikow and A. Stranieri</i>	336
Scaling Up the Rule Generation of C4.5 <i>Z. Zheng</i>	348
Data Mining Based on the Generalization Distribution Table and Rough Sets <i>N. Zhong, J. Dong and S. Ohsuga</i>	360

Posters

Constructing Personalized Information Agents <i>C.-H. Chang and C.-C. Hsu</i>	374
Towards Real Time Discovery from Distributed Information Sources <i>V. Cho and B. Wüthrich</i>	376
Constructing Conceptual Scales in Formal Concept Analysis <i>R. Cole, P. Eklund and D. Walker</i>	378
The Hunter and the Hunted – Modelling the Relationship Between Web Pages and Search Engines <i>D. L. Dowe, L. Allison and G. Pringle</i>	380
An Efficient Global Discretization Method <i>K. M. Ho and P. D. Scott</i>	383
Learning User Preferences on the WEB <i>F. Jacquet and P. Brenot</i>	385
Using Rough Sets for Knowledge Discovery in the Development of a Decision Support System for Issuing Smog Alerts <i>I. Jagielska</i>	388
Empirical Results on Data Dimensionality Reduction Using the Divided Self-Organizing Map <i>T. Koshizuka, H. Ogawa and J. Fulcher</i>	390

Mining Association Rules with Linguistic Cloud Models <i>D. Li, K. Di, D. Li, and X. Shi</i>	392
A Data Mining Approach for Query Refinement <i>Y. Liu, H. Chen, J. X. Yu and N. Ohbo</i>	394
CFMD: A Conflict-Free Multivariate Discretization Algorithm <i>Y. Lu, H. Liu and C. L. Tan</i>	397
Characteristic Rule Induction Algorithm for Data Mining <i>A. Maeda, H. Maki and H. Akimori</i>	399
Data-Mining Massive Time Series Astronomical Data Sets – A Case Study <i>M.K. Ng, Z. Huang, and M. Hegland</i>	401
Multiple Databases, Partial Reasoning, and Knowledge Discovery <i>C. Nowak</i>	403
Design Recovery with Data Mining Techniques <i>C. Monks de Oca and D. L. Carver</i>	405
The CLARET Algorithm <i>A. R. Pearce and T. Caelli</i>	407
LR Tree: A Hybrid Technique for Classifying Myocardial Infarction Data Containing Unknown Attribute Values <i>C. L. Tsien, S. M. Hamish, S. F. Fraser and I. S. Kohane</i>	409
Modelling Decision Tables from Data <i>G. Wets, J. Vanthienen and H. Timmermans</i>	412
A Classification and Relationship Extraction Scheme for Relational Databases Based on Fuzzy Logic <i>M. Vazirgiannis</i>	414
Mining Association Rules for Estimation and Prediction <i>T. Washio, H. Matsuura and H. Motoda</i>	417
Rule Generalization by Condition Combination <i>H. Xue and Q. Cai</i>	420
Author Index	423

Knowledge Acquisition for Goal Prediction in a Multi-user Adventure Game

D.W. Albrecht, A.E. Nicholson and I. Zukerman

Department of Computer Science, Monash University
Clayton, VICTORIA 3168, Australia
phone: +61 3 9905-5225 fax: +61 3 9905-5146
`{dwa,annn,ingrid}@cs.monash.edu.au`

Abstract. We present an approach to goal recognition which uses a Dynamic Belief Network to represent domain features needed to identify users' goals and plans. Different network structures have been developed, and their conditional probability distributions have been automatically acquired from training data. These networks show a high degree of accuracy in predicting users' goals. Our approach allows the use of incomplete, sparse and noisy data during both training and testing. We then apply simple learning techniques to learn significant actions in the domain. This speeds up the performance of the most promising dynamic belief networks without loss in predictive accuracy.

1 Introduction

In this paper, we apply a Dynamic Belief Network (DBN) formalism for knowledge representation and reasoning to predict users' goals, actions and locations in an adventure game called the "Shattered Worlds" Multi-User Dungeon (MUD) (Section 2), and compare the effect of different DBNs on our system's predictive ability (Section 3). We also discuss the application of simple machine learning techniques to learn significant actions in the domain; this results in a speed up in performance without loss in predictive accuracy (Section 4).

2 The Domain

The MUD has over 4,700 locations, over 7,200 actions, and 20 different quests (goals). The plan recognition problem in the MUD involves anticipating a player's quest as early as possible, as well as anticipating his/her next action and next location continuously. This problem is exacerbated by the presence of typographical errors, spelling mistakes, newly defined commands and abbreviations for commands. Quests may involve solving non-trivial puzzles or interacting with various MUD characters. Players usually know which quest they wish to achieve, but they don't always know which actions are required to complete a quest. In addition, they often engage in activities that are not related to the completion of a specific quest, such as chatting with other players. As a result, players typically perform between 25 and 500 actions until they complete a quest, even though only a fraction of these actions may actually be required to achieve a quest.

Table 1. Sample data for the Avatar quest

Action No.	Time	Player	Location	Action
1	773335156	spillage	room/city/inn	ENTERS
12	773335264	spillage	players/paladin/room/trading_post	buy
17	773335291	spillage	players/paladin/room/western_gate	bribe
28	773335343	spillage	players/paladin/room/abby/guardhouse	kill
37	773335435	spillage	players/paladin/room/abby/stores	search
40	773335451	spillage	players/paladin/room/shrine/Billy	worship
54	773335558	spillage	players/paladin/room/brooksmith	give
60	773335593	spillage	players/paladin/room/shrine/Dredd	avenger
62	773335596	spillage	players/paladin/room/abby/chamber	Avatar quest

Analysis of the MUD yields the following features: (1) it is extremely difficult to obtain a perspicuous representation of the domain (the effects and preconditions of many actions are not fully known); (2) there may be more than one way to achieve a goal; (3) some sequences of actions may lead to more than one eventual goal; (4) some actions leading to a goal must be performed in sequence, while other actions are order-independent; (5) users may interleave the actions performed to achieve two or more goals or may perform actions that are not related to any domain goal; (6) the states of the MUD are only partially observable: the only information available at present is a user's actions (obtained from the user's keyboard commands), a user's locations (obtained from the system), and a few system messages; and (7) the outcome of the actions is uncertain.

The MUD software collects the actions performed by each player, the visited locations and the quest instances each player completed. A DBN is then constructed on the basis of the collected data (Section 3). In the current implementation, each data point is composed of: (1) a time stamp, (2) the name of the player, (3) the number of the login session, (4) the location where the action was executed, and (5) the name of the action. Table 1 illustrates some of the 62 actions performed by the spillage character to achieve the Avatar quest (the number of the login session is not shown).¹ Without domain knowledge, it is extremely difficult to determine by inspection which of these actions are necessary to complete the quest, the order of the necessary actions, or whether an action had the intended outcome.

3 Belief Updating

In the next four subsections, we develop DBN models for the MUD domain. To this effect, we perform the following actions: (1) identify the interesting domain variables which become the network nodes (Section 3.1); (2) consider dependencies between the domain variables and the manner in which the values of these variables change over time (these dependencies correspond to several alternative network structures, which we

¹ Note that the first and last entries in the Action column of the table are not actions done by the player, but system messages stating that a player has entered the MUD and that a quest has been completed, respectively. Also, at present, the MUD software does not record keyboard commands regarding an agent's movements on the horizontal plane. In addition, only the first word of each command is considered during training and testing.

investigate experimentally) (Section 3.2); (3) describe how the Conditional Probability Distributions (CPDs) are constructed from the collected MUD data and how we handle actions, locations and quests which do not occur in the training data (Section 3.3); and (4) present the data processing algorithm and belief update equations (Section 3.4). In Section 3.5 we discuss the experimental results obtained with the different DBNs.

3.1 Network Nodes

Based on the data we have for the MUD domain, the domain variables, which are represented as nodes in the DBNs, are as follows:

Action (A): This variable represents the possible actions a player may perform in the MUD, which we take to be the first string of non-blank characters entered by a user, plus the special *other* action, which includes all previously unseen actions. The results given in Section 3.5 are obtained with a state space size, $|A|$, of 4,904 actions. This state space is reduced from the original space of 7,259 actions, since this research takes into account only those runs where a quest was completed. In Section 4 we consider the effect of taking into account only significant actions.

Location (L): This variable represents the possible locations of a player, plus the special *other* location, which includes all previously unseen locations. The results given in Section 3.5 are obtained with a state space size, $|L|$, of 3,369 locations. As for actions, this state space is reduced from the original space of 4,722 locations.

Quest (Q): This variable represents the 20 different quests a player may undertake, plus the *other* quest, which includes all quests not seen in the training data, and the *null* quest. The variable representing the previous quest achieved is set to *null* if the user has just started a session.

3.2 Network Structure

Several DBN models involving these nodes were investigated (Figure 1). These models are not pure DBNs; the changes in actions and locations over time are represented, but it is assumed that a player's current quest, Q' , does not change and depends on the previous quest, Q .

Figure 1(a) shows the most complex of these models (called *mainModel*). This model stipulates that the location L_i at time step i depends on the current quest, Q' , and the previous location at time step $i - 1$, and that the action A_i depends on the previous action, the current quest and the current location.² These dependencies are based on the following observations and assumptions. The dependence of a location on the previous location reflects the physical limitations of the domain, whereby most movement is to a topologically adjacent location (although teleporting is also possible). The dependence of an action on the previous action reflects the assumption that there is some correlation between pairs of actions; clearly, there may be longer sequences of actions which are connected, but including these dependencies in the model would defeat the Markovian assumption inherent in DBNs, which in turn would cause an explosion in the state space of the problem. The dependence of an action on a location reflects the observation

² Models where locations depend on actions were not considered, because, as stated above, the MUD software keeps only partial records of actions which cause changes in locations.

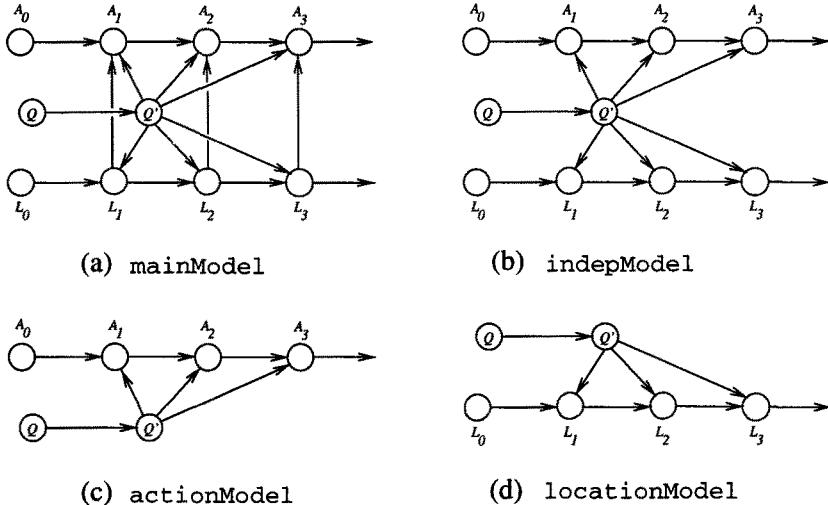


Fig.1. Dynamic Belief Networks for the MUD: (a) mainModel; (b) indepModel; (c) actionModel; (d) locationModel.

that certain actions are mainly performed in certain locations, e.g., objects are usually bought at the market and food consumed at the inn. The dependence of both location and action on the current quest reflects the assumption that most quests are completed in a particular subset of locations by undertaking particular actions.

The model in Figure 1(b) (called `indepModel`) relaxes the direct dependence between actions and locations, assuming that given the current quest, the current action and location are independent. Finally, the models in Figure 1(c) and 1(d) (called `actionModel` and `locationModel` respectively) are simpler still; they take into consideration either actions or locations in relation to quests (but not both).

3.3 Probability Acquisition

The CPDs are constructed automatically from the collected MUD data as follows. The data is pre-processed to take the following form:

Previous Quest	Current Quest	Current Action	Current Location	Next Action	Next Location
null	teddy	scream	room/sewer/sewer20	u	room/city/alley1

A frequency count is maintained for each entry in the CPDs that was observed during training. These entries represent action, location and quest combinations that are relevant to the belief update formulas presented in Section 3.4 and were seen during training. In order to account for missing data, that is, the possible actions, locations and quests that do not occur in the training data, we adjust the frequencies so that the resulting CPDs include some probability that the other value may occur. This adjustment consists of adding a small number that corresponds to Good's *flattening constant* (Good, 1965) to the non-zero frequencies. A factor of 0.5, which is computed by the Minimum Message Length theory (Wallace, 1990) assuming the prior is constant on seen events, was used for the results obtained in this paper (other flattening constants are discussed in (Albrecht et al., 1997)). The frequencies are then converted into CPDs.

-
1. Receive initial data:
PreviousQuest, PreviousAction, PreviousLocation.
 2. Add data as evidence for nodes Q , A_0 and L_0 .
 3. Update belief on nodes Q' , A_1 and L_1 .
 4. Loop from $n = 1$ until quest is achieved
 - 4.1 Receive new data: Action, Location.
 - 4.2 Add data as evidence for nodes A_n and L_n .
 - 4.3 Update belief on nodes Q' , A_{n+1} and L_{n+1} .
 - 4.4 $n = n + 1$.
-

Fig. 2. Belief update algorithm for processing a run.

3.4 Belief Updating

Once a DBN is constructed, new data from a user is added to the network as evidence, and belief is updated regarding that user's next action and next location and the current quest being undertaken.

A *run* is a sequence of action-location pairs, beginning either after a player enters the MUD or after a player completes the previous quest, and ending when a new quest is achieved. The belief update algorithm applied for processing a run is given in Figure 2. If the run begins when a player enters the MUD, PreviousQuest, PreviousAction and PreviousLocation are set to null. In this case, a value of null is added as evidence for time-step 0 to nodes Q , A_0 and L_0 . Otherwise (the run begins upon completion of a quest), the last quest completed, last action performed and last location visited are used. The evidence nodes for the domain at time-step $n + 1$ are: the last completed quest, Q , the previous actions, A_0, \dots, A_n , and the previous locations, L_0, \dots, L_n .

There are underlying loops in the network structures shown in Figure 1, such as the loop between A_i , A_{i+1} and Q' in Figures 1(a), 1(b) and 1(c) and the loop between L_i , L_{i+1} and Q' in Figures 1(a), 1(b) and 1(d). This would seem to indicate that we must use an inference algorithm based on clustering, conditioning or stochastic simulation (Pearl, 1988). However, further analysis of these structures, together with the location of the evidence nodes, identifies d-separations (Pearl, 1988), indicating that certain nodes are conditionally independent (see (Albrecht et al., 1997) for details of this analysis). Using these independence relations, we obtain the following belief update equations for mainModel corresponding to Steps 3 and 4.3 in the belief update algorithm (the simplified update equations for indepModel, actionModel and locationModel appear in (Albrecht et al., 1997)).

Step 3 (time-step 0):

$$\Pr(Q' = q'|q, a_0, l_0) = \Pr(Q' = q'|q),$$

$$\Pr(L_1 = l_1|q, a_0, l_0) = \sum_{q'} \Pr(L_1 = l_1|l_0, q') \Pr(q'|q, a_0, l_0),$$

$$\Pr(A_1 = a_1|q, a_0, l_0) = \sum_{q', l_1} \Pr(A_1 = a_1|a_0, l_1, q') \Pr(l_1|l_0, q') \Pr(q'|q, a_0, l_0).$$

Step 4.3 (time-step n+1):

$$\begin{aligned}
 \Pr(Q' = q' | q, a_0, l_0, \dots, a_{n+1}, l_{n+1}) &= \\
 \alpha \Pr(l_{n+1} | l_n, q') \Pr(a_{n+1} | a_n, l_{n+1}, q') \Pr(Q' = q' | q, a_0, l_0, \dots, a_n, l_n), \\
 \Pr(L_{n+1} = l_{n+1} | q, a_0, l_0, \dots, a_n, l_n) &= \\
 \sum_{q'} \Pr(L_{n+1} = l_{n+1} | l_n, q') \Pr(q' | q, a_0, l_0, \dots, a_n, l_n), \\
 \Pr(A_{n+1} = a_{n+1} | q, a_0, l_0, \dots, a_n, l_n) &= \\
 \sum_{q', l_{n+1}} \Pr(A_{n+1} = a_{n+1} | a_n, l_{n+1}, q') \Pr(l_{n+1} | l_n, q') \Pr(q' | q, a_0, l_0, \dots, a_n, l_n),
 \end{aligned}$$

where α is a normalizing constant.

The update equations for time-step $n + 1$ show that the new belief for the current quest (Q'), the next action (A_{n+1}) and the next location (L_{n+1}) can be computed from the previous beliefs in the values for the current quest and the CPD entries for the latest evidence received. The evidence before this, i.e., the evidence for action nodes A_0, \dots, A_{n-1} and location nodes L_0, \dots, L_{n-1} , does not have to be considered explicitly, having been “folded” into the beliefs for A_n and L_n .

3.5 Experimental Results

In this section we present empirical results showing how the DBN models described in Section 3.2 predict current quest, next action and next location. We quantitatively compare the different models in terms of their quest, action and location predictions using the average predicted probability of the actual quest, action and location.³ All the results presented in this section were obtained by choosing randomly 80% of the 3,017 quest-achieving runs in our corpus as training data, and using the remaining runs as test data, with cross-validation using 20 different splits of the data.⁴ During each test run, we used the belief update algorithm shown in Figure 2.

Average prediction of a domain variable, i.e., current quest, next action or next location, is the average across all test runs of the predicted probability of this variable at each point during the performance of a quest:

average prediction = $\frac{1}{n} \sum_{i=1}^n \Pr(\text{actual value of } \textit{variable} \text{ in the } i\text{-th test run}),$
 where *variable* may be either current quest, next action or next location, and n is the number of test runs performed.

In order to compare across runs where the number of recorded actions varies, we use the percentage of actions taken to complete a quest. That is, we apply our measures of performance at 0.1%, 0.2%, 0.3%, ..., 1%, 1.1%, ..., 2%, 2.1%, ..., 100% of quest completion. These percentages are computed so that there is at least one data point for the quest with the largest number of actions; the quests with only a few actions will have a single action that corresponds to several data points.

³ In (Albrecht et al., 1997) we consider another measure for comparing the performance of our models, and examine the effect of varying the size of the training set on quest predictions.

⁴ During testing, a value which was not seen in the training data gets classified as other.

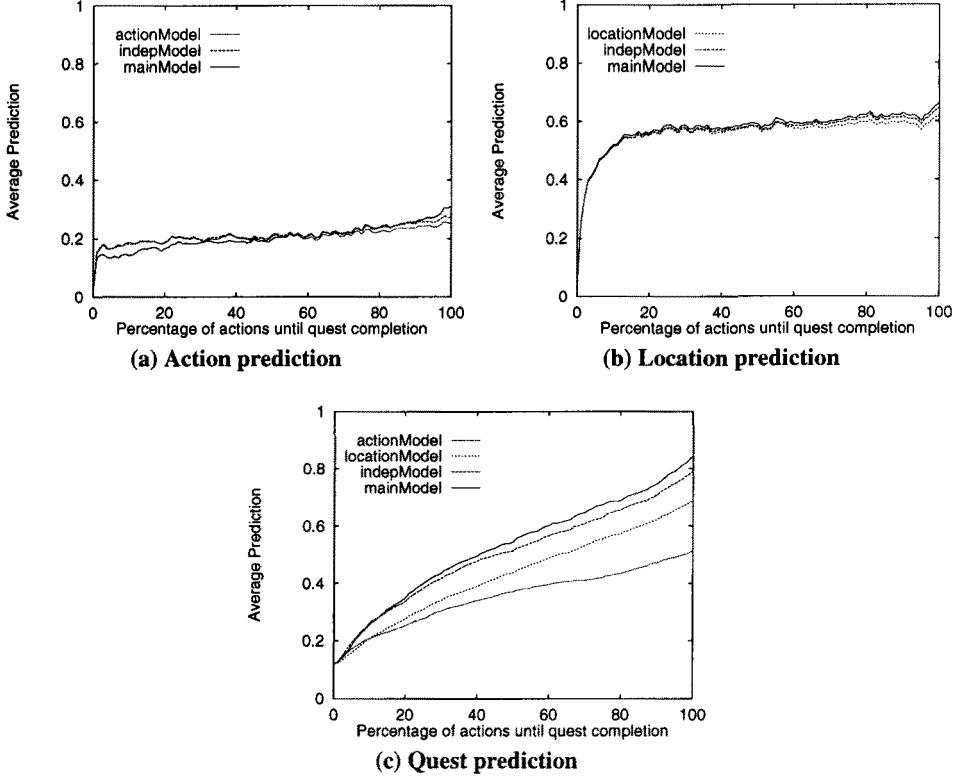


Fig. 3. Average prediction of models for (a) actions, (b) locations, and (c) quests.

Figure 3(a-c) shows the average predictions for actual actions, locations and quests for the four models. The null hypothesis that there is no significant difference between the models' predictions was tested using a T-test with 38 d.f. for each set of predictions and each pair of models.⁵ Figure 3(a) shows that given the previous quest and current action, we can predict the next action with an average probability around 0.2. It also shows that the average predictions of `indepModel` are virtually indistinguishable from those of `actionModel` until about 46% of the actions have been completed (at the 0.5% significance level). According to Figure 3(a), the average predictions of `mainModel` improve steadily from being worse than those of the other two models (for the first 52% of a run), to being indistinguishable from the average predictions of `actionModel` (at 53% of a run, when `indepModel` gives better average predictions), and finally to being the highest (after 92% of a run has been completed). In contrast, the relative performance of `actionModel` starts dropping off after 60% of the actions have been performed. The T-tests confirmed these observations at the 0.5% significance level.

⁵ We have 38 degrees of freedom because each of the two averages and standard deviations are computed from 20 splits; one degree of freedom is deducted because we use the sample standard deviation when calculating the difference between the two models, yielding $2 \times (20 \text{ splits} - 1 \text{ standard deviation}) = 38$.

In Figure 3(b) we see that given the previous quest and current location, we can predict the next location with an average probability of about 0.6. The best predictions are produced by `mainModel1`, with the predictions produced by `indepModel` being marginally lower, and those produced by `locationModel` being slightly lower again. The T-tests show at the 0.5% significance level that after 50% of the actions have been performed, the average location predictions obtained by `mainModel1` are better than those of the other models.

Figure 3(c) shows that given the previous quest, current action and current location, the average prediction for the next quest rises steadily from 0.12 to about 0.83 for `mainModel1`, which gives the best average predictions. The average predictions obtained with `indepModel1` are slightly lower, while those obtained with `actionModel1` and `locationModel1` are significantly lower. The T-tests show at the 0.5% significance level that the average predictions of `mainModel1` are significantly better than those of the other models after only 20% of the actions have been performed.

Selecting the best model. The selection of a model is based on the quality of its predictions of the variables of interest, that is, current quest, next action and next location. The meaning of the average prediction measure in terms of the predictive ability of a model is as follows. Given that a model has yielded an average prediction probability p for the actual value of a particular variable at a certain point in time, if we perform our prediction for this variable at that point in time by randomly selecting a value based on the calculated probability, then on average our prediction will be correct $100p\%$ of the time. For example, according to Figure 3(c), if we are asked to predict the current quest based on the average prediction obtained by `mainModel1` after 80% of the actions have been performed, then our prediction will be correct on average about 69% of the time. The results in Figure 3 indicate that the average action predictions of `mainModel1` are worse for about half a run, while its quest predictions are consistently better; `indepModel1` performs well for all three predictions. The average location predictions obtained by `locationModel1` are comparable to those obtained by `mainModel1` and `indepModel1`, but its average quest predictions are substantially lower. Thus, only `indepModel1` and `mainModel1` have been retained for the remainder of our analysis.

4 Learning the language

We apply simple models and update equations to our domain because the size of the domain compounds the complexity of the problem. Another type of simplification involves reducing the size of the state space representation by ignoring non-significant actions in the domain. In this section we describe how to apply simple machine learning techniques to achieve this simplification, and present experimental results.

As indicated in Section 2, the presence of non-significant actions, such as typographical errors and spelling mistakes, exacerbates the plan recognition problem in the MUD. While the basic approach certainly handles this sort of noisy data, these actions increase the number of actions that must be dealt with during training and testing without actually having an impact on the states of the MUD. In order to overcome the difficulties caused by these extraneous actions we used a Minimum Message Length (MML) classifier (Wallace, 1990) to learn the language that is understood by the MUD

Table 2. Classes of actions in the MUD

Class	% of total	# of times action was performed	# of players who performed action	Sample commands
		Range % of class	Range % of class	
C9	65.6%	1-2 (99.16%)	1 (100.00%)	guard ..will 1- 23e
C6	21.9%	2-7 (97.01%)	1-2 (100.00%)	101: tell 1move I've
C10	6.7%	7-54 (97.55%)	2-7 (98.70%)	abuse alis copy kill
C11	4.5%	20-1,096 (97.33%)	20-54 (95.20%)	break dance free pray
C8	1.3%	148-22,026 (98.94%)	54-403 (98.26%)	answer climb sell shout

(details on how the classifier is used are given in (Albrecht et al., 1997)). We then considered only the actions in this language both during training and testing of our DBN models. The results obtained with this classifier were used in the two best models discussed in Section 3: `indepModel` and `mainModel`.

4.1 Classification

Only two attributes were given to the classifier: (1) how many times each action was performed, and (2) how many players performed it. The classifier was run with all our data, and it identified five classes. Table 1 shows sample actions in these classes together with the percentage of the actions contained in each class, the ranges which contain most of the attribute values in each class (for each of the attributes), and the percentage of the elements in each class accounted for by these ranges. For example, class C9 contains 65.6% of all the actions typed in by players; these actions were normally typed in only once or twice (99.16% of the actions in this class were typed in once or twice), and were executed by one player only (all the actions in this class were executed by a single player).⁶ As can be seen in Table 1, classes C8 and C11 contain the most widely used actions that were typed in by the largest number of players. The actions in these classes are ‘sensible’ MUD domain actions and communication actions. In contrast, classes C9 and C6 contain infrequent actions used by a few players. These are typographical and spelling errors, personal aliases, infrequent numerical combinations, and words used in snippets of conversations. Class C10 contains a mixture of actions of the type found in C8 and C11 (but less frequently used), and actions of the type found in C6 and C9 (but more frequently used). Thus, the classifier has identified two classes of actions that are unlikely to be in the MUD language, namely C6 and C9, and two classes of actions that are very likely to be in the MUD language, namely C8 and C11. However, the situation is not so clear with respect to C10.

In order to determine whether this action-screening process is useful in general, and whether C10 should be included in the MUD language, we trained and tested `indepModel` and `mainModel` with two candidate MUD languages: the language composed of the actions in C8 and C11 (called C8.11); and the language composed of the actions in C8, C10 and C11 (called C8.10.11). Language C8.11 reduces the action space from 4,904 actions to 415, while C8.10.11 reduces the action space to 926 actions. In

⁶ The classifier generates numerical identifiers for its classes as it creates them. When classes are merged, they are not re-numbered, hence the resultant non-consecutive numerical identifiers.

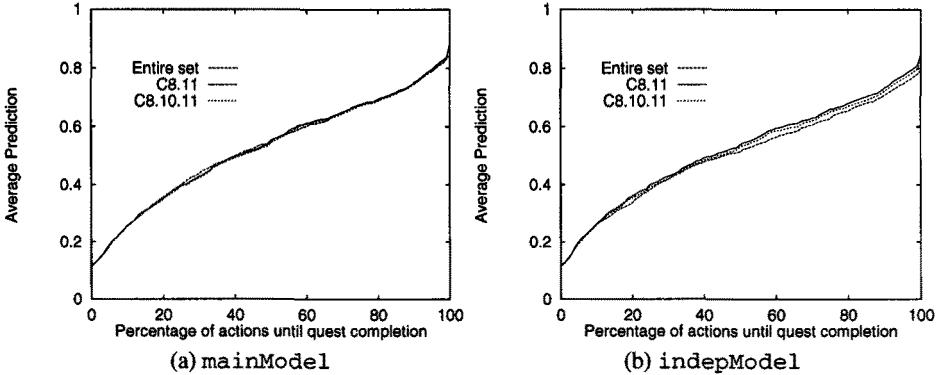


Fig. 4. Average quest predictions for `mainModel` and `indepModel` when trained with C8.11, C8.10.11 and the entire action set.

order to obtain a preliminary indication of the validity of the learned languages, we checked how many consecutive non-significant actions (i.e., actions outside these languages) are typically performed. This test is based on the notion that if the ignored actions are mainly typographical and spelling errors, typically there should be short sequences of these actions, since a player would immediately correct an erroneous command with a correct (significant) command. Indeed, the average number of consecutive non-significant actions is 1.89 for C8.11 and 1.24 for C8.10.11, thereby supporting our hypothesis that the actions excluded from C8.11 and C8.10.11 are not significant.

4.2 Results

As stated above, DBN training and testing was performed with the actions in the language, rather than with the entire action set. Training was performed with 80% of the data, and testing with 20% cross-validated using 20 different splits (while language learning was performed with the entire data set). During both training and testing, the non-significant actions are simply ignored, i.e., they are removed from the data set. This means that the DBN does not learn to predict the occurrence of non-significant actions, and that the performance of such actions by a player does not affect the predictions made by the plan recognizer during testing. This causes difficulties when trying to compare the performance of DBNs which use different languages, such as C8.11, C8.10.11 and the entire action set, for action and location predictions. Hence, we compare the performance of these DBNs only for quest predictions. Figure 4(a-b) shows the average predictions of `mainModel` and `indepModel` when trained and tested on C8.11, C8.10.11 and the entire action set.

According to Figure 4, the performance of `mainModel` when trained with each of the training sets is slightly better than that of `indepModel`. `indepModel` obtained the best average predictions when trained and tested with C8.11. After 38% of a run the average predictions obtained with C8.11 are significantly better than those obtained with C8.10.11, and after only 13% of a run the average predictions obtained with C8.11 are significantly better than those obtained with the entire data set. The average predictions obtained by `mainModel` when trained and tested with the entire data set, C8.11 and

C8.10.11 are virtually indistinguishable from each other for most of a run. For small portions of a run the average predictions obtained with the entire data set are better (between 26% and 33% and between 41% and 49%) and for other portions they are worse (between 56% and 65%). These results indicate that training and testing with C8.11 had more impact on the results obtained with `indepModel` than on those obtained with `mainModel`. This can be explained by the observation that in `mainModel` the link between locations and actions lowers the probabilities of non-significant actions, which in turn reduces their contribution to quest predictions. Therefore, the removal of non-significant actions does not substantially change quest predictions. The absence of this link in `indepModel` (which results in higher probabilities for non-significant actions) means that non-significant actions interfere more with quest predictions. Hence, their removal has a higher impact on quest predictions.

Interestingly, the memory requirements of `mainModel` (which involve representing the CPDs) were reduced only by about 8% when trained and tested with the smallest language, C8.11. The reason for this relatively small reduction is that the CPDs for `mainModel` were initially very sparse (and zero probability events were not explicitly represented). Further, the screening of non-significant actions may introduce previously unseen action-action combinations, which must be represented in the CPDs. The reduction in memory requirements for `indepModel` was about 15%. It is worth noting that the reduction in memory requirements for both models was similar in absolute terms, indicating that similar information was removed from both models when trained with C8.11. However, in terms of percentages, the reduction is higher for `indepModel` since its CPDs are smaller than those of `mainModel`. In contrast, there was a considerable reduction in computation time during training and testing for both `mainModel` and `indepModel` (training and testing for `mainModel`, which has the highest computational requirements, went down from about one day to about half a day). The reductions obtained during training took place when constructing the CPD tables used in the prediction of next action and next location; the reductions obtained during testing took place when making these predictions.

5 Discussion

We use Dynamic Belief Networks (DBNs) for knowledge representation and reasoning. We have presented and compared four DBNs which predict a user's next location, next action and current quest based on a training corpus. The structure of the networks is based on our domain knowledge; we have proposed four basic network structures that model different underlying dependencies in the domain. In addition, there is an automated knowledge acquisition aspect to our approach, whereby the conditional probability tables in the networks are constructed from the collected MUD data. Simple models are required since the number of possible values for each node makes network training and evaluation a computationally complex task.

The comparison between the four presented models gives some insight into the underlying dependencies in the domain. The accuracy of quest predictions obtained when using both a user's locations and a user's actions is significantly higher than the accuracy of the predictions obtained when using actions or locations alone. Our results further indicate that the system's belief regarding which quest a user is attempting affects both

location and action predictions. Interestingly, `mainModel` performs the worst on action predictions for most of a run, as measured by the average prediction. As indicated in Section 4.2, this may be due to the link from locations to actions in `mainModel`, which reduces the probability of non-significant actions (these actions form a large percentage of the observed actions). Another contributing factor may be the increased size of the CPD table used in `mainModel` and the resulting sparseness of that table; the lack of a sufficient number of data points results in undue emphasis being placed upon a relatively small number of observations. There is little difference between the location predictions of `locationModel`, `mainModel` and `indepModel`. Hence, if the focus is on location predictions only, the simplest model, i.e., `locationModel`, should be used.

An important feature of our approach is that due to the probabilistic training, predictions are based on actions that are *normally* performed to achieve a goal, rather than on actions that necessarily advance a user towards the achievement of a goal. Thus, actions that are necessary to achieve a goal (and hence performed by a large number of users to satisfy a particular goal) have a relatively significant effect on predictions. On the other hand, actions that are performed across many goals and extraneous actions (i.e., those which do not contribute to the achievement of any goal, such as typographical errors), have little effect on the prediction of a particular goal.

In an extension to the basic approach, we have attempted to screen out these extraneous actions using an MML classifier. By learning the language of the MUD, we have substantially reduced the computation time required by our DBNs while gaining quest predictive power (the reductions in memory requirements were rather modest).

To summarize, if we wish to predict all the domain variables of interest, that is, current quest, next action and next location, the best models are `mainModel` or `indepModel`. However, if we are interested only in quest predictions, `mainModel` obtains the highest average predictions when trained on the actions in the MUD language C8.11 (these training sets also reduce the computational requirements of `mainModel`).

Acknowledgments

This research was supported in part by grant A49600323 from the Australian Research Council. The authors thank Michael McGaughey for writing the data collection programs for the MUD, and Ariel Bud for valuable contributions throughout this project.

References

- Albrecht, D. W., Zukerman, I., and Nicholson, A. E. (1997). Bayesian models for key-hole plan recognition in an adventure game (extended version). Technical Report 328, Department of Computer Science, Monash University, Victoria, Australia.
- Good, I. J. (1965). *The Estimation of Probabilities: An Essay on Modern Bayesian Methods*. Research Monograph No. 30. Cambridge, Massachusetts: MIT Press.
- Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems*. San Mateo, California: Morgan Kaufmann Publishers.
- Wallace, C. (1990). Classification by minimum-message-length inference. In Goos, G., and Hartmanis, J., eds., *ICCI '90 – Advances in Computing and Information*. Berlin: Springer-Verlag. 72–81.

Hybrid Data Mining Systems: The Next Generation

Sarabjot S. Anand, John G. Hughes
Northern Ireland Knowledge Engineering Laboratory
University of Ulster at Jordanstown,
Northern Ireland BT37 0QB
E-mail: {ss.anand,jg.hughes}@ulst.ac.uk

Abstract

The promise of Hybrid Systems as the next generation of Data Mining systems is investigated. This work is motivated by the obvious limitations of paradigms for Data Mining that are being used in isolation. We present a classification of hybrid systems based on the level of interaction between the component paradigms and present example objectives for the development of such systems. We highlight possible hybrid solutions by discussing various hybrid systems that may be developed to enhance the Nearest Neighbour algorithm. These include statistical measures to enhance distance measures, a loose coupling with Neural Networks or alternatively a tight coupling with genetic algorithms, to discover attribute weights. We also establish enhancements to the k-NN that make it appropriate for use as a paradigm for addressing regression data mining goals. We provide results obtained using these systems, comparing them with more traditional paradigms used to solve regression goals within Data Mining.

1 Introduction

Data Mining is a fast growing technology born through the synergy of existing technologies like machine learning, database technology, statistics, high performance computing, visualisation and mathematics. It is defined as "the semi-automated process of extracting previously unknown, potentially useful and interesting knowledge from large and often disparate, historical data sources" [ANAN97b]. The objective of current Data Mining research is to address the numerous challenges posed by the data being large, historical and disparate as well as addressing functionality requirements of the users such as data pre-processing, knowledge post-processing and incorporation of domain expertise.

Data Mining is not only widely used in the sense of the sectors of industry where it has been applied but also in its applicability to a wide range of tasks. The tasks, referred to as "Data Mining Goals", have been classified into nine classes: Classification, Regression, Discovery of Associations, Discovery of Sequential Patterns, Temporal Modelling, Deviation Detection, Dependency Modelling, Clustering and Characteristic Rule Discovery. For each class there are a number of paradigms that can be used to achieve the goal at hand. For example to solve a classification goal we may use Neural Networks, Rule Induction, Decision Tree Induction, Genetic Algorithms and Nearest Neighbour techniques. Two main considerations for choosing a paradigm are accuracy and perspicuity. Each paradigm has its own set of biases and the decision on which paradigm to use for a particular problem is normally based on empirical studies. However, empirical evaluations of the various paradigms are biased by the expertise resident in the data mining expert involved in the project. Thus, we suggest that the major challenges of data mining are: attempting to remove biases that exist in the various paradigms using hybrid

paradigms and removing the user element within the setting of parameters that affect performance.

A number of these challenges can be addressed by using hybrid systems for Data Mining. Biases may be removed by combining the paradigm with another that balances biased aspects of the original paradigm. Also, biases may be removed by tailoring paradigms normally used to achieve one data mining goal for use in another data mining goal. In section 2 we discuss a framework for developing Hybrid Systems for Data Mining. We provide a classification of hybrid systems and suggest various challenges that can be addressed by the different classes of hybrid systems. In section 3 we discuss the development of a hybrid system that transforms the original k-NN algorithm into a more robust, unbiased, noise tolerant paradigm that may be used for classification as well as regression tasks. We conclude this paper by discussing future enhancements to Data Mining systems that employ hybrid paradigms.

2 A Framework for Developing Hybrid Systems

Hybrid Systems are not new to the field of AI [GOON95]. Goonatilake et al. define an intelligent hybrid system as a system where two or more techniques are combined in a manner that overcomes the limitations of individual techniques. They classify hybrid systems into three types: Function Replacing systems, Inter Communicating systems and Polymorphic systems.

Function Replacing systems are those in which shortcomings of a technique are overcome using another technique that is particularly strong in that aspect of information processing. Shortcomings may be in terms of performance, reliability or missing functionality. For example, fuzzy systems lack learning power, which may be overcome by using a hybrid Neuro-fuzzy system. Inter Communicating systems are systems where no single technique is applicable for solving a problem due to its complexity in terms of the multiplicity of sub-tasks. Systems that are enhanced to exhibit multiple information processing capabilities within a single architecture are known as Polymorphic systems e.g. using neural networks for symbolic processing.

Within Data Mining, a hybrid system may be used for addressing a multiplicity of objectives, enhancing functionality of existing systems, improving performance, increasing robustness and decreasing or balancing biases. Each of these objectives may be met by using Function replacement, Inter communicating or Polymorphic systems. In fact, the distinction between Polymorphic and Function Replacement systems is blurred since Polymorphism is normally achieved through function replacement. Thus, we base our classification of Hybrid Systems for Data Mining on the extent to which the various participating techniques interact with each other. We define two classes of Hybrid Systems: Loosely-coupled and Tightly-coupled Hybrid Systems.

Loosely coupled hybrids consist of a number of techniques that need to be employed to solve a complex problem due to the multiplicity of application tasks or to improve performance of the task. For example, the cross-sales problem consists of a number of sub-tasks, two of which were identified during the Problem Specification phase of the

Data Mining Process [ANAN98a], as Data Mining Tasks (DMT). While one of the DMTs was identified as having a characteristic rule discovery goal, the other was identified as having a deviation detection goal. Thus, a solution could only be provided by using a loosely coupled hybrid of a characteristic rule discovery algorithm and a deviation detection algorithm. The use of sampling in association rules discovery is an example of a hybrid system that was developed with performance enhancement as the objective [TOIV96]. Components of a loosely coupled hybrid may execute in parallel or sequentially and communicate through inter-component messages. This class of hybrids corresponds to Goonatilake et al.'s Inter-Communicating class of hybrids.

Tightly coupled hybrids consist of an integration of more than one technique to solve a single task. The objective of this hybridisation may be: to add missing functionality, to provide mechanisms for coping with new tasks, increasing robustness or reducing unrealistic biases. The hybrid systems using the Nearest Neighbour algorithm (k-NN), described in the next section, and implemented within the Mining Kernel System [ANAN97a], contain examples that address a number of these objectives. It uses statistical techniques for removing biases within the distance metric used by the k-NN algorithm, a genetic algorithm for discovering attribute weights automatically (add missing functionality and increasing robustness), an adaptation of traditional prediction functions to allow k-NN to be used for regression Data Mining goals (coping with new tasks). We also investigate a loosely coupled hybrid of the k-NN algorithm with a Neural Network as an alternative to the genetic algorithm for discovering attribute weights.

3 Hybrid Systems using the Nearest Neighbour Algorithm

The k nearest neighbour algorithm (k-NN) is one in a family of distance based algorithms. The basic model consists of a set of exemplars, a distance metric, a prediction technique once the nearest neighbours have been retrieved and weightings for attributes and exemplars. The training phase of the k-NN algorithm is the discovery of attribute and exemplar weights. In this section we discuss each of these aspects of the model and training used by k-NN, detailing biases and suggesting remedies through the development of hybrid systems.

3.1 The Exemplar Set

Normally k-NN is used to solve classification problems. The exemplars, in their simplest form, are a set of (attribute, value) pairs for each of the independent attributes and a classification label. We used k-NN on two regression problems: predicting survival in months for colorectal cancer patients and predicting house prices. The cancer data set consisted of fifteen clinico-pathological features, five of which were categorical, the rest being continuous or ordered. The housing data set consisted of ten independent variables, four of which were categorical and the rest were continuous. Table 1 gives example data from the cancer exemplar set used to evaluate the hybrid system developed. Due to space limitation we limit our discussion to the cancer exemplar set. Adapting k-NN for use in regression goals requires new functionality within the distance metric and prediction technique aspects of the model.

The question arises, however, as to why k-NN must be adapted for use within regression as opposed to using regression tree induction or neural networks that are the standard techniques used to achieve regression goals within Data Mining. Regression trees are known to not perform well in sparse data sets. Also, tree induction algorithms commonly restrain the search for models to those that split the search space using partitions that are parallel to the axes. In cases where attributes within the data set are correlated, such induction algorithms result in complex trees and less accurate models. On the other hand, while neural networks are less brittle, they are opaque structures lacking in perspicuity. In the medical as well as land valuation sectors, the ability to explain decisions based on a set of exemplars is attractive. Thus, k-NN is potentially the paradigm of choice. However, it has not been used, widely, to solve regression problems in the past. Thus, the development of a hybrid system is a potential solution.

Table 1 Example Cancer Data

Sex	Path Type	Polarity	Configuration C	Pattern	Infiltration	Fibrosis	Mitotic Count	Penetration	Differentiation	Dukes Stage	Age	Obstruction	Site	Survival	
2	1	4	2	2	6	4	4	3	3	2	4	79	1	7	1
2	1	6	3	4	2	4	4	4	4	4	5	77	0	7	1
4	1	6	3	4	6	6	5	3	5	6	6	53	1	5	2
2	1	2	2	2	4	2	4	6	4	2	4	82	1	7	2
2	1	6	3	4	4	6	5	5	4	4	6	47	1	4	2

3.2 Distances Metrics

The general form of the distance metric used in k-NN is defined as:

$$dist(v_1, v_2) = \left(\sum_{i=1}^n d_i^r \right)^{1/r}$$

where, v_1 and v_2 are data tuples

$$\text{and } d_i = \begin{cases} |v_1^i - v_2^i| & \text{if the } i\text{th attribute is continuous} \\ 1 & \text{if the } i\text{th attribute is categorical and } v_1^i \neq v_2^i \\ 0 & \text{if the } i\text{th attribute is categorical and } v_1^i = v_2^i \end{cases}$$

The most commonly used distance metrics for k-NN are the Euclidean distance and Manhattan distance correspond to $r = 2$ and 1 respectively in the formula above.

Given a target case, in data sets with a mix of categorical and real valued attributes, the Manhattan and Euclidean distance metrics bias retrieval to those exemplars that match the values of the categorical attribute values of the target case. This bias, as we shall see, is not necessary and enhanced distance metrics can be developed to remove this bias.

A number of metrics have been suggested in the literature to remove the bias within traditional distance metrics. However, they are appropriate only when the dependent attribute is categorical. Most significant among these is the Valued Difference Metric [STAN86]. In cases where the dependent attribute is continuous, VDM cannot be used. However, using the same basic principle of comparing the effect of different values of the independent categorical attribute on the dependent attribute's distribution is used to develop enhanced distance metrics.

The values of an independent categorical variable define a partition of the data set. Each value of the variable thus has a distribution of the dependent variable associated with it. The basic idea behind the enhanced distance metrics introduced here is that the distance between any pair of values of the independent variable should depend on the difference between the distributions of the dependent variable conditioned on these values. One way of comparing two distributions is to compare how the various statistics of central tendencies differ. Alternatively, techniques that use the complete distribution itself as opposed to predefined statistics of the distribution may be used [PRES92].

Three different approaches were investigated based on central tendency statistics. These were Mean-based distance, Coefficient of Variation based distance, Significant Mean based distance.

The Mean-based distance measure maps the values of the categorical attribute (V_C) onto the interval $[0, 1]$ using the following mapping, i :

$$i: V_C \rightarrow [0,1] \\ \text{such that}$$

$$i(v_c^k) = \frac{\overline{x_o(v_c^k)} - \min\{\overline{x_o(v_c^j)} : 0 \leq j \leq m\}}{\max\{\overline{x_o(v_c^j)} : 0 \leq j \leq m\} - \min\{\overline{x_o(v_c^j)} : 0 \leq j \leq m\}}$$

where, m is the cardinality of V_C

and $\overline{x_o(v_c^j)}$ is the mean of the distribution of the dependent attribute O conditioned on value v_c^j of V_C

A well-known problem of using means in this manner is the fact that the mean of the distribution is very sensitive to any outliers. This needs to be taken into account, either by using statistical techniques for identifying and removing outliers or by taking the deviation into account in the measure used for comparing the two distributions. One such measure is the coefficient of variation, which is calculated as the ratio of the standard deviation of the distribution to its mean. The coefficient of variation based distance metric also maps the values of the categorical attribute onto the interval $[0, 1]$ using the mapping.

$$i: V_C \rightarrow [0,1] \\ \text{such that}$$

$$i(v_c^k) = \frac{\overline{cv_o(v_c^k)} - \min\{\overline{cv_o(v_c^j)} : 0 \leq j \leq m\}}{\max\{\overline{cv_o(v_c^j)} : 0 \leq j \leq m\} - \min\{\overline{cv_o(v_c^j)} : 0 \leq j \leq m\}}$$

where, m is the cardinality of V_C

and $\overline{cv_o(v_c^j)}$ is the coefficient of variation of the distribution of the dependent attribute O conditioned on value v_c^j of V_C

Table 2 shows the mapping of values of the categorical variable 'Site', in the colorectal cancer data set onto the [0, 1] interval using both metrics

Table 2 Mean-based mapping of the Site attribute values

Value	Mean	Mapped Value	Coefficient of Variation	Mapped Value
Sigmoid colon	36.57	0.51	0.87	0.78
Splenic flexure	16.17	0	0.8	0.63
Transverse colon	25.57	0.23	0.96	0.97
Descending colon	54.18	0.96	0.76	0.55
rectum	42.88	0.67	0.7	0.427
Caecum	35.78	0.5	0.97	1
Hepatic flexure	45.32	0.74	0.69	0.406
Ascending colon	55.48	1	0.498	0

The third method uses the Student's t-test for analysing the significance of the difference in mean of the distributions.

A small value of the probability of 't' implies a large significance in the difference in means. Thus, $1 - \text{prob}(t)$ is used as the distance metric.

Using this metric the distance matrix in Table 3 was calculated for the Site variable of the cancer data set. Note that only half of the matrix is shown, as it is symmetric by definition.

Table 3: Distance matrix for Site variable using the Significant Mean measure

Site colon		Splenic flexure		Trans. colon		Desc. colon		Rectum		Caecum		Hepatic flexure	
Splenic flexure		0.375		Trans. colon		Desc. colon		Rectum		Caecum		Hepatic flexure	
Trans. colon		0.322		0.209		0.23		0.33		0.073		0.37	
Desc. colon		0.379		0.345		0.23		0.33		0.073		0.37	
rectum		0.315		0.45		0.45		0.21		0.37		0.134	
Caecum		0.03		0.33		0.29		0.39		0.68		0.406	
Hepatic flexure		0.056		0.071		0.322		0.23		0.427		0.63	
Ascending colon		0.43		0.53		0.67		0.05		0.78		0.97	

In the case of the Mean and Coefficient of Variation, the continuous values that the categorical values are mapped to are used to calculate the Euclidean distance. For example:

$d_{\text{SITE}}(\text{Splenic flexure}, \text{Transverse Colon}) = 0.23 - 0 = 0.23$ using the mean-based mapping and $d_{\text{SITE}}(\text{Splenic flexure}, \text{Transverse Colon}) = 1 - 0.457 = 0.543$ using the co-efficient of variation-based mapping.

Note that if there were only two unique values of a categorical variable both these mappings would yield the same results as would be obtained using the basic Euclidean or Manhattan distance measures, since the two categorical values would be mapped onto 0 and 1.

When using the Significant Mean based metric, the value $d_{\text{SITE}}(i, j)$ is calculated by looking up the (i, j) th element in the distance matrix constructed. Unlike the mean-based and co-efficient of variation mappings, when using the Significant Mean measure, even when there are only two values of the categorical variable, the distance is based on the significance of the difference observed between the means of the distributions of the output variable corresponding to these values. For example, using the significant mean based measure $d_{\text{SEX}}(\text{Male}, \text{Female}) = 0.37$ as opposed to 1.

Figure 1 shows the percentage matches of each of the attributes in the colorectal cancer data set, using cross-validation and retrieving 5 nearest neighbours for each target. In the cancer data set, attributes 1, 2, 5, 14 and 15 are categorical. As can be seen from these figures, the new distance metrics remove the bias towards matches within the categorical attributes.

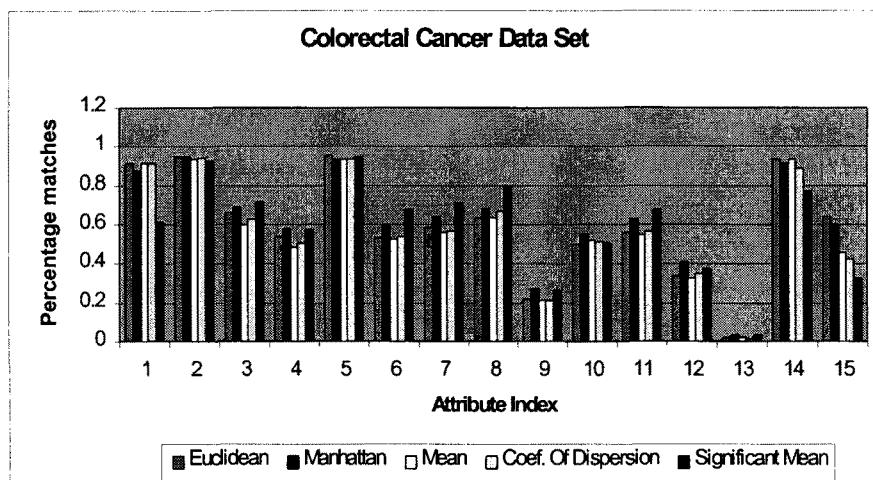


Figure 1: Percentage Matches of Attributes in the exemplar set

3.3 Prediction Technique

Once the nearest neighbours have been retrieved they must be utilised in some way to output a prediction for the dependent attribute. Common techniques used by k-NN algorithms include the most common value (MCV) and voting based prediction (VBP). In MCV the output attribute's value that is most common amongst the k nearest neighbours retrieved is chosen as the class to which the target case is classified. The disadvantage of this technique is the fact that it does not take the

distance metric into account when arriving at the final prediction. The VBP technique rectifies this by allocating votes to each of the retrieved nearest neighbours based on their distance from the target case. The smaller the distance, the greater the vote. The output class that accrues the maximum votes is the final prediction.

In the case of the dependent attribute being continuous, VBP can be easily extended and defined as the weighted sum of the output attribute values of the nearest neighbours is used. The weights are based on the distance of the neighbours from the target case. Once again, the smaller the distance, the greater the weight used.

A parameter affecting the prediction phase of the k-NN is the value of 'k'. Clearly, if k is 1, the prediction for the target case is the value of the dependent attribute for the retrieved neighbour. As 'k' increases, the tolerance of k-NN to noise increases. However, for very large values of 'k', the accuracy obtained is poor as over-generalisation occurs and the nearest neighbours retrieved belong to different concept clusters, leading to errors in prediction.

3.4 Discovery of Attribute Weights

The distance metric, or similarity function, is sensitive to irrelevant, interacting and noisy attributes [WETT95]. Thus, in the presence of such 'rogue' attributes, using the k-NN algorithm leads to high error rates. Variants of the k-NN algorithm use weighted distance metrics, assigning a weight to each of the attributes in the data set. A large weight is associated with highly relevant attributes and a low weight with irrelevant attributes. Wettschereck [WETT96] identified five dimensions along which techniques for attribute weight selection can be measured. These are Bias, Weight Space, Representation, Generality and Knowledge. The *Bias* dimension has two possible values, *Performance* and *Preset*. Performance techniques are similar in philosophy to Wrapper methods for feature selection [KOHA95]. These techniques use the k-NN algorithm as part of the evaluation function used in the search for optimal attribute weightings. Hybrids developed for this purpose are tightly coupled. On the other hand Preset methods, similar to filter methods for feature subset selection, arrive at weightings independently of algorithm performance and hybrid systems developed for this purpose are loosely coupled. Along the *Weight Space* dimension, attribute weighting techniques are classified into *Continuous* and *Binary* techniques. This classification is based on whether the attribute weights are binary (i.e. 0 or 1) or continuous. Using Binary techniques is equivalent to attribute selection as opposed to attribute weighting. Within the *Representation* dimension, techniques can either be *Given* or *Transformed*. Classification along this dimension is based on whether the technique outputs weightings for attributes within the original data set as opposed to any transformation of the data set. Transformations that may be carried out include binarisation of categorical variables and transforming attributes into principal components. Converting categorical variables into a set of binary variables has the ill effect of increasing dimensionality, increasing performance overheads on the attribute weighting selection algorithm. Also, individual weightings for each of the categorical values need to be combined heuristically to compare the relevance of the categorical variable as a whole with other attributes within the data set. A point to

note, however, is that the enhanced distance metrics discussed earlier in this paper are equivalent to Preset methods for assigning weights to the set of binary attributes that the categorical attribute would be transformed into, without the disadvantage of ‘dimensionality explosion’. Within the *Generality* dimension, techniques that assign global weights are known as *Global* techniques while weights that assign different weights to attributes in different parts of the attribute space are called *Local* techniques. Along the knowledge dimension, techniques may be *Intensive* or *Poor* based on whether or not the assignment of weights utilised domain knowledge.

Three techniques were investigated to assign weights to attributes:

- Acquiring weights from the domain expert
- Using a Neural Network for discovering attribute significance and transforming them into weights
- Using a genetic algorithm for discovering attribute weights

Note that other techniques suggested in literature like those based on Mutual Information [DAEL92] and Incremental Hill Climbers [SALZ91] cannot be used here as they require the dependent attribute to be categorical i.e. they are only appropriate when using the k-NN algorithm for classification tasks, whereas our use of k-NN is for building a regression model.

Acquisition of weights from the domain expert is clearly knowledge intensive as the weights are based solely on domain knowledge. It is clearly Preset as performance and biases of the algorithm being used for model building are not taken into account when selecting the weights. The weights assigned by the domain expert belonged to the interval $[0, 1]$ and are continuous. Also, weights were assigned to the attributes globally i.e. the weights were constant throughout the attribute space. No transformations were carried out on the data before assigning weights. Disadvantages of this technique are that the attribute weights tended to be subjective to a large extent, i.e. while the domain expert had a fair idea of the order of significance of the different attributes within the data set, the assigning of specific weights often did not prove to be optimal.

The second technique was a loosely coupled hybrid of a multi-layered perceptron using a constructive learning algorithm with the k-NN algorithm. While this technique did take the performance of the Neural Network into account when attributing significance to the various attributes, the assignment of weights was in complete isolation from the k-NN algorithm which was ultimately used to build the model. Thus we classify this technique as Preset along the bias dimension. Weights are calculated using the formula:

$$w_j = \frac{s_j - \min\{s_k : 1 \leq k \leq n\}}{\max\{s_k : 1 \leq k \leq n\} - \min\{s_k : 1 \leq k \leq n\}}$$

where, s_k is the significance of the k th attribute
and n is the number of attributes in the data set

By definition, the attribute weightings calculated using the formula above are continuous and belong to the interval [0,1]. The Neural Network implementation used in the study converts categorical input attributes into a set of binary inputs. However, the value of significance is calculated for the categorical attribute as a whole. The effect of the intermediate transformation needs further investigation. The weights arrived at are global in nature and as the neural network uses no domain knowledge in arriving at the weights, this technique is classified as Poor along the Knowledge dimension. Disadvantages of using this technique are the fact that the attribute significance arrived at varied with the choice of Neural Network parameter settings and that the attribute weights arrived at did not take biases of the k-NN algorithm into account. Also, in effect, the user has a number of extra parameters to set that pertain to the Neural Network.

The third technique investigated is a tightly coupled hybrid system that uses a genetic algorithm in conjunction with the k-NN algorithm. The Genetic Algorithm is used to carry out a directed search in the space of attribute weightings. The k-NN algorithm with 10-fold cross validation is used as the evaluation function for calculating fitness for the individual chromosomes within the genetic population. Genetic Algorithms through an exploration (using crossover and mutation genetic operators) coupled with exploitation (through survival of the fittest based selection) search mechanism have proven to be a robust search paradigm that tends to avoid local minima provided its parameters are chosen with care. The simple genetic algorithm has a number of parameters that need to be selected: a number of coding schemes for chromosomes that form the genetic population exist, the size of the genetic population, the mutation rate, crossover rate and type of crossover operator used. In our case the chromosome consists of a number of genes. Each gene is a binary code representing a single attribute weight. The number of genes that form a chromosome is equal to the number of independent attributes within the exemplar set. A seven-bit gene size was chosen so that a three-digit precision for attribute weights may be achieved.

The weights assigned to the attributes are clearly continuous to a 3-digit degree of precision thus, the technique is classified as continuous along the Weight Space dimension. Also, no transformations are carried out on the underlying data set as the bias towards matching categorical variables has already been dealt with in section 3.2. Thus, for the attribute weighting technique used, there is no difference between categorical and numeric attributes. Thus, this technique is classified as 'Given', along the Representation dimension. The weights are assigned globally using this technique and no domain knowledge is utilised in the assignment of weights apart from performance feedback from the k-NN algorithm used in calculating fitness of the genetic population. Thus this technique is classified as 'Global' along the Generality dimension and 'Poor' along the Knowledge dimension. The main disadvantage of this technique is the additional set of genetic parameters that need to be set by the user.

Table 4 shows different attribute weightings arrived at using the different attribute weighting techniques. Note that the attribute weights shown in the table above obtained using the neural network and genetic algorithm are the best set of weights obtained through empirical studies with varying parameters settings for the neural

network and genetic algorithm. Table 5 compares the accuracy of the various hybrid approaches with the accuracy of traditional regression approaches. As can be seen the GA-kNN hybrid provides the best results using the Significant Mean based distance metric.

Table 4 Attribute Weights discovered for Colorectal Cancer

Technique	Sex	Pain Type	Polarity	Configuration	Pattern	Infiltration	Erosion	Venoous Invasion	Mitotic Count	Penetration	Differentiation	Dukes Stage	Age	Obstruction	Site
Expert	0.05	0.39	0.39	0.72	0.5	0.5	0.33	0.61	0.28	0.67	0.72	0.89	1	0.33	0.33
NN	0.24	0.43	0.3	0.47	0.08	0.21	0.4	0.2	0	0.46	0	1	0.56	0.62	0.33
GA	0.03	0.83	0.00	0.03	0.34	0.66	0.81	0.22	0.95	0.19	0.29	0.75	0.68	0.67	0.14

Table 5 Comparison of Accuracy of various systems for regression

Technique	Mean Absolute Error (Months)
Base Line Prediction (Using Survival Mean as prediction)	27.59
Regression Tree Induction	34.69
Neural Networks	24.8
Unweighted k-NN/ Significant-Mean based distance	24.73
k-NN with Domain Expert's weights/ Significant-Mean based distance	23.64
k-NN/ Neural Nets Hybrid/ Significant-Mean based distance	23.82
k-NN/ GA Hybrid/ Significant-Mean based distance	20.89

For a complete discussion on the use of the hybrid systems described in this paper for prognosis of colorectal cancer refer to [ANAN98].

These results, firstly, prove the validity of pursuing the development of hybrid systems for Data Mining. Secondly, it reiterates the findings of Wettschereck et al. [WETT97] that Performance based techniques are more accurate than Preset techniques for discovery of attribute weightings for k-NN. Thirdly, they prove the validity of pursuing the goal of enhancing the biased traditional distance metrics within k-NN. These results while clearly encouraging, are by no means an end in themselves. Future work that needs carried out is briefly described in the next section.

4 Conclusions

The use of hybrid systems for Data Mining was investigated. We described a number of challenges of Data Mining that can be addressed by using hybrid systems. We provided a classification of hybrid systems based on the amount of interaction between the component paradigms. We then designed hybrid systems that utilised the k-NN algorithm in conjunction with statistical techniques, neural networks and genetic algorithms to overcome a number of deficiencies traditionally associated with the k-NN algorithm. The results obtained improved upon results obtained using the k-NN, neural network and regression tree in isolation, providing evidence of the promise of hybrid systems within Data Mining. This empirical proof has further

strengthened the authors' view that the next generation of Data Mining systems will necessarily consist of hybrid solutions.

Further work that is being undertaken within the authors' laboratory is two pronged. Firstly, specific extensions are proposed to the k-NN based hybrid, based on distribution comparison techniques for enhanced distance metrics, extending attribute weighting techniques to local techniques possibly using exception spaces as a starting point. On a more general note, particular challenges that need addressed are the implications of developing hybrid systems on the Data Mining Process and development of techniques to reduce user intervention at the level of parameter setting. The development of such Adaptive Data Mining systems is according to the authors, the final frontier of Data Mining.

5 Acknowledgements

The authors would like to thank Peter Hamilton of Queen's University of Belfast for providing the colorectal cancer data set and William McCluskey from the School of Built Environment, University of Ulster for making available the housing data set.

6 References

- [ANAN97a] S. S. Anand, B. W. Scotney, M. G. Tan, S. I. McClean, D. A. Bell, J. G. Hughes, I. C. Magill. Designing a Kernel for Data Mining, IEEE Expert, 12 (2), pp. 65 - 74, 1997.
- [ANAN97b] S. S. Anand, A. G. Buchner. Decision Support using Data Mining, (to be published), Financial Times Management (ISBN: 0-273-63269-8), April, 1998.
- [ANAN98] S. S. Anand, P. W. Hamilton, A. S. Smith, J. G. Hughes. Intelligent Systems for Prognosis of Colorectal Cancer Patients, To appear in Proc. of CESA'98 Special Session on Intelligent Prognostic Methods, April, 1998 <http://inchinn.infj.ulst.ac.uk/htdocs/reports.html>.
- [ANAN98a] S. S. Anand, A. R. Patrick, J. G. Hughes, D. A. Bell. A Methodology for Cross Sales using Data Mining, to appear in the Knowledge Based Systems Journal, 1998.
- [DAEL92] W. Daelemans, A. van den Bosch. Generalisation performance of backpropagation learning on a syllabification task., Proc. of TWLT3, 1992.
- [GOON95] S. Goonatilake, S. Khebbal. Intelligent Hybrid Systems, John Wiley and Sons, 1995.
- [KOHA95] R. Kohavi, D. Sommerfeld. Feature Subset Selection Using the Wrapper Method: Overfitting and Dynamic Search Space Topology, Proc. of the International Conference on Knowledge Discovery and Data Mining, 192 - 197, 1996.
- [PRES92] W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery. Numerical Recipes in C, Second Edition, Cambridge University Press, 1992.
- [SALZ91] S. L. Salzberg. A nearest hyperrectangle learning method, Machine Learning, 6, pp. 251 - 276, 1991.
- [STAN86] C. Stanfill, D. Waltz. Towards memory-based reasoning. Communications of the ACM, Vol. 29, pp. 1213 – 1228, 1986.
- [TOIV96] H. Toivonen. Sampling Large Databases for Association Rules, Proc. of 22nd International Conference on Very Large Databases, pp. 134 - 145, Mumbai, India
- [WETT97] D. Wettschereck, D. W. Aha, T. Mohri. A Review and Empirical Evaluation of Feature Weighting Methods for a Class of Lazy Learning Algorithms, Artificial Intelligence Review, Volume 11, Issues 1-5, pages 273-314, Special Issue on Lazy Learning, April 1997.
- [WETT95] D. Wettschereck, D. W. Aha. Weighting features. In Proceedings of the First International Conference on Case-Based Reasoning, Lisbon, Portugal, 1995.

Discovering Case Knowledge Using Data Mining

S. S. Anand[†], D. Patterson[†], J. G. Hughes[†], D.A.Bell[‡]

[†] Northern Ireland Knowledge Engineering Laboratory

[‡] School of Information and Software Engineering
University of Ulster, Newtownabbey, County Antrim

Northern Ireland BT37 0QB

E-mail: {ss.anand, wd.patterson, jg.hughes, da.bell}@ulst.ac.uk

Tel: 44-1232-366671

Fax: 44-1232-366068

Abstract

The use of Data Mining in removing current bottlenecks within Case-based Reasoning (CBR) systems is investigated along with the possible role of CBR in providing a knowledge management back-end to current Data Mining systems. In particular, this paper discusses the use of Data Mining in two aspects of the M² system [ANAN97a], namely, the acquisition of cases and discovery of adaptation knowledge. We discuss, in detail, the approach taken to discover cases and outline the methodology to discover adaptation knowledge. For case discovery, a Kohonen network is used to identify initial clusters within the database. These clusters are then analysed using C4.5 and non-unique clusters are grouped to form concepts. A regression tree induction algorithm is then used to ensure that the concepts are rich in information required to predict the dependent variable in the data set. Cases are then chosen from each of the identified concepts as well as outliers in the database. Initial results obtained in the acquisition of cases are presented and analysed. They indicate that the proposed approach achieves a high reduction in the size of the case base.

1 Introduction

Recent interest in Data Mining [FAYY96] has led to a flurry of new technologies based on an amalgamation of techniques from statistics, machine learning, database technology and high performance computing. However, while most emphasis has been placed on the development of new discovery algorithms, few techniques have been suggested in the literature in relation to the management of the discovered knowledge. Knowledge post-processing techniques suggested in the literature have concentrated on filtering the discovered knowledge retaining only that knowledge which is interesting [GEBH94] while knowledge management has been largely ignored.

In this paper we discuss the use of Data Mining in a novel role of a back-end technology for Case-Based Reasoning (CBR) systems. However, we suggest that the benefits of this role of Data Mining has the potential of not only improving our ability to build CBR systems but could also provide a basis for knowledge management in Data Mining systems as a whole. This work continues the work being carried out on the M² CBR System currently under development in the authors laboratory [ANAN97a] and concentrates specifically on identifying seed cases within a domain with which to initially populate a case base. This has long been identified as a bottleneck in CBR development with many systems either simply using all available cases [ASHL88] or relying on an expert to identify key cases. The former methodology introduces unacceptably high overheads in terms of storage and retrieval

while the latter relies heavily on good human judgement and is very time consuming. We propose that Data Mining could play a central role in identifying the seed cases automatically from a database of operational data. Additionally we propose that it could also be utilised to discover adaptation rules used for adapting retrieved cases. In this paper we discuss initial results obtained in the use of Data Mining for case and adaptation knowledge discovery using a hybrid methodology utilising a number of different Data Mining paradigms.

Reducing the problem of knowledge acquisition by opting for large case bases, where likelihood of retrieving a near-exact match is high and the need for adaptation is low, leads to additional problems. Recent research [SMYT95] has shown that such CBR systems suffer from the *swamping problem*. This is the problem associated with searching large case bases for the appropriate cases to solve the current problem. The swamping problem is a special case of the *utility problem*, which occurs when the cost associated with searching for relevant knowledge outweighs the benefit of applying the knowledge. Smyth et al. showed that a smaller case base that preserves the competence of the CBR system is a preferred state.

CBR has been proposed as a paradigm for Data Mining however, a number of factors reduce its applicability in a number of domains. For example, the knowledge acquisition bottleneck associated with the construction of cases and the assignment of relevance weights to the case features. Also, in most cases, the use of CBR is limited to user-driven induction of decision trees and the use of the nearest neighbour algorithm with domain expertise being incorporated in the process [CURE96]. In this paper we take the view that rather than being a paradigm for Data Mining, CBR is in fact a useful end product of Data Mining, providing a knowledge representation, manipulation and reasoning facility. Thus, in this paper we not only address the use of Data Mining as a method of overcoming a number of short comings in present CBR systems but also as a means to address the issue of knowledge post-processing within Data Mining. Little attention has been paid to this aspect of Data Mining in literature, though it has been recognised as an important challenge [FAYY96].

Thus, the proposal within this paper may be summarised as, the transition from a large operational database, which is of minimal use in decision making, to a substantially smaller semantically rich case base. This case base includes a set of adaptation knowledge in addition to specific cases that empower decision makers with a central knowledge resource that can be utilised in strategic decision support. Recent developments in Data Mining have made it possible to envisage this transition with minimal user intervention. Optionally, the domain expert may choose to add his/her domain expertise to the CBR system in the form of case representation and structuring, domain or background knowledge or the definition of complex, possibly linguistic features [DUBI97, SHUS97]. An additional benefit in utilising CBR as the resulting knowledge representation and manipulation component of a Data Mining system is its intuitive appeal and ability to model human cognition, making it attractive to end users.

The rest of the paper is in the following format. We first discuss in Section 2, relevant pieces of related research that have been adapted and enhanced, to meet our needs within the M² system. We then describe the process of case knowledge

discovery in Sections 3 and 4, relating the various paradigms of Data Mining utilised for this purpose. While Section 3 describes the process of acquiring cases from operational databases, Section 4 concentrates on the discovery of adaptation knowledge. We provide preliminary results from the housing domain to make the case knowledge acquisition process explicit. We conclude the paper in Section 5 by discussing future work that the authors propose to undertake, to further the results presented in this paper.

2 Related Work

There are two aspects of CBR research that are most relevant and related to parts of the M² system. These are, the work presented by Hanney et al. [HANN96] on discovering adaptation rules from the case base and the work presented by Smyth et al. [SMYT95] on learning to forget cases. We now discuss this work and contrast it to our approach to using Data Mining within the M² system.

Hanney et al. presented a technique based on discovering association rules from the case base and generalising the rules discovered whenever required using the *closing interval rule* [MICH83]. They use case-pair comparison to discover the adaptation rules. However, the number of rules that can possibly be discovered using an association algorithm is enormous. Thus, they employ two heuristics to reduce the number of discoverable rules. Firstly, they use the similarity measures to select case-pairs to be used for rule generation. Secondly they suggest the deletion of duplicate rules. They also investigate the use of domain knowledge in the form of known adaptation rules, irrelevant features, contextual dependencies and feature interaction.

The work presented in this paper differs from that presented by Hanney et al. in a number of ways. Firstly, while Hanney et al. consider the case base to be the whole database, we take the view that the database and the case base are two distinct entities. While the case base consists of distinct concept instances, the database contains all instances of problems occurring in the problem space. Secondly, we see the use of case-pair comparisons as the source of rules an unnecessary overload as it leads to the discovery of very specific rules that are highly context dependent and require accurate domain knowledge to be available. We discuss our approach to adaptation knowledge discovery in Section 4.

Smyth et al. presented two new approaches for avoiding the swamping problem by deleting cases based on their *coverage* and *reachability*. Together, the coverage and *reachability* provide a measure of the *dispensability* of the case from the case base. The *coverage* of a case is defined as the set of target cases that it can be adapted to solve while *reachability* of a case is the set of cases in the case base that can be adapted to solve that case. In general, as the space of target cases is immense, the *coverage* and *reachability* are estimated by using the case base as a representative sample of the target space. Based on these measures, Smyth et al. classified cases within the case base into four groups: Pivotal (if its *reachability* is a singleton consisting of itself), Auxiliary (if its *coverage* is subsumed by the *coverage* of a case to which it is *reachable*), Spanning (if its *coverage* space links regions covered by other cases) and Support (groups of cases having the same *coverage*). The deletion policy suggested by Smyth et al. was to delete Auxiliary cases first, then support

cases, then spanning cases and finally pivotal cases. If more than one case is a candidate for deletion, the utility of each case is taken into account when deciding on which case to delete. Two observations about the method suggested here are that the *coverage* and *reachability* of a case is dependent on the adaptation knowledge available and secondly, it is unclear as to why a pivotal case would ever need to be deleted.

In the case of the M² system, the case base is being built from scratch as opposed to the whole database being considered identical to the case base. Thus, it is not a question of deleting cases from the case base but a question of how to select the most appropriate cases from the database that should be added to the case base. However, we use the definitions of Pivotal and Auxiliary cases to discover cases for the case base from large databases. We discuss our approach to case selection in Section 3.

3 The Case Discovery Process

In this section we discuss the process followed within the M² system to discover cases from large operational databases. The discovery of the more general adaptation knowledge is postponed to the next section.

In our discussion of the methodology used for case identification and discovery of adaptation knowledge we use data from the housing domain. This data was collated over a one year period from sales of houses in the Belfast area of Northern Ireland. It consisted of a total of 883 tuples. Twelve attributes were stored in the data set for each house. These were: Quarter of the year the sale took place, House Type (Detached, Apartment etc.), Age of the house (New, post 1950, 1920 -1950 etc.), Floor area, Number of bedrooms, Number of reception rooms, Zip code, Bathroom inside the house (Boolean Y/N), Garage (Boolean Y/N), Oil fired central heating (Boolean Y/N), Need of repair (Boolean Y/N) and Sale price.

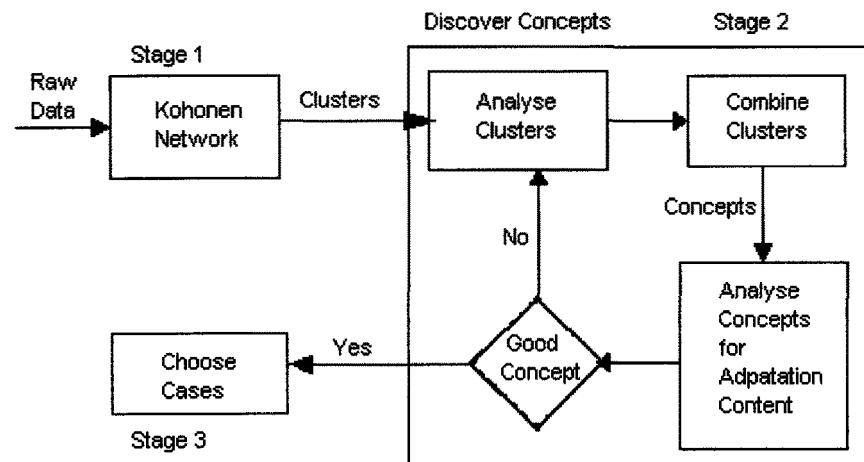


Figure 1 : Schematic Representation of Case Discovery Process

The first step in the process of discovering cases is to cluster the database. The clusters identified are considered to be different concepts within the case base. Each of these concepts identifies a subset of the problem domain that differs characteristically from the rest of the problem domain [DUBI96, DUBI96a, SMIT81]. For example, in the housing domain, it may be the case that pricing of flats is different from that of bungalows or it may be that the domain subsets are more specific than that. It is possible that the relevant attributes in each of the identified concepts is different and therefore the cases created for each of these concepts would have a different structure, with irrelevant attributes removed from each *concept case base*. The domain expertise provided by the domain expert may now be used to map the database records belonging to the concept into the *concept case space*.

Figure 1 shows the various stages of the process of discovering concepts and, consequently, cases from a large operational database. The implementations of the Kohonen network, C4.5 and Regression tree induction algorithm used here are part of the CLEMENTINE Data Mining software [CLEM96]. For discovering the concepts within the Housing database, a Kohonen network was used (Stage 1). Kohonen networks require the user to set a number of parameters that could potentially have a profound effect on the quality of the clustering. These are the number of clusters (rows and columns within the two-dimensional map that the input data must be mapped onto by the network), the learning rates of the two phases within the Kohonen networks learning stage and the number of learning cycles in each phase. Taking into consideration the size of the data set we were dealing with we chose the number of clusters for our exercise as 16 providing for an average of approximately 55 tuples per cluster. Values greater than 16 would have resulted in an unnecessary

spread of the tuples into meaningless clusters. We trained the network for a number of different cycle settings taking the default setting for the learning rate parameter 'eta' (Phase 1: 0.3 and Phase 2: 0.1). We observed that the clustering produced by the Kohonen network seemed to stabilise after training had been carried out for approximately two hours. Unfortunately, there are no known stopping criteria that would indicate convergence of the Kohonen network as in the case of Multi-layered perceptrons and so one must rely of empirical results and heuristics. Figure 2 shows the clustering obtained from the Kohonen network. Numbers shown in the figure are the number of records grouped into each cluster i.e. Cluster 00 has 69 records and cluster 32 has 56 records.

Figure 2: Sample Output from Kohonen Network

	0	1	2	3
0	69	82	70	99
1	57	11	07	35
2	39	16	18	64
3	96	37	56	127

Having identified the clusters in the data, the next stage is to investigate these clusters to investigate how distinct these clusters really were. During this stage we used the following heuristic: *The clusters are distinct if it is possible to discern, to a high level of accuracy, the membership of the tuples falling into the different clusters*. Thus, we used the C4.5 algorithm to try to induce a decision tree with the dependent variable as the cluster label to which each tuple in the data set belongs. We used the *holdout sample* technique for validating the decision tree obtained from C4.5. One-

Having identified the clusters in the data, the next stage is to investigate these clusters to investigate how distinct these clusters really were. During this stage we used the following heuristic: *The clusters are distinct if it is possible to discern, to a high level of accuracy, the membership of the tuples falling into the different clusters*. Thus, we used the C4.5 algorithm to try to induce a decision tree with the dependent variable as the cluster label to which each tuple in the data set belongs. We used the *holdout sample* technique for validating the decision tree obtained from C4.5. One-

third of the complete data set constituted the holdout sample while two-thirds was utilised as the training set. The results of the validation of the decision tree are shown in Table 1.

	00	01	10	11	13	02	20	21	22	23	03	30	31	32	33
00	17	4	1	0	0	0	0	0	0	0	0	0	0	0	0
01	3	22	1	1	0	1	0	0	0	0	1	0	0	0	0
10	0	0	16	0	0	0	0	0	0	0	0	0	0	0	0
11	1	0	4	0	0	0	0	0	0	0	0	0	0	0	0
12	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	11	0	0	0	0	0	0	0	0	0	0
02	0	0	2	1	0	19	0	0	0	0	0	0	0	0	0
20	2	0	0	0	1	0	5	0	0	0	0	4	0	0	0
21	0	0	0	0	0	0	0	1	1	0	0	0	0	1	0
22	0	0	0	0	0	0	0	0	2	3	0	0	0	1	0
23	0	0	0	0	2	0	0	0	3	15	0	0	2	1	1
03	0	0	1	0	1	0	0	0	0	0	28	0	0	0	0
30	1	0	0	0	0	0	1	1	2	0	0	25	0	1	0
31	0	0	0	0	0	0	0	2	2	0	0	7	6	2	0
32	0	0	0	0	0	0	0	0	1	0	0	0	0	12	5
33	0	0	0	0	0	0	0	0	0	0	0	0	0	4	40

Table 1: Matrix showing the predicted clusters (y-axis) against the actual clusters (x-axis) for each record in the test data set

As can be seen from Table 1, the original clusters were not completely distinct. The cells marked in bold font represent the obvious groups of clusters that appeared to be distinct.

These similar clusters were combined into one *concept* where applicable and any outliers were removed. For example concept 1 now includes clusters 0 and 1, concept 2 comprises of clusters 10 and 11, concept 3 comprises of cluster 13, concept 4 consists of cluster 2, concept 5 includes clusters 22 and 23, concept 6 comprises of clusters 20, 30 and 31, concept 7 comprises clusters 32 and 33 and concept 8 consists of cluster 3. Clusters 12 and 21 were identified as outliers and tuples belonging to these clusters are added to the case base as individual cases.

Thus, 16 original clusters were reduced to 8 distinct concepts.

	1	2	3	4	5	6	7	8
1	47	1	0	1	0	0	0	1
2	3	17	0	0	0	0	0	0
3	0	0	11	0	0	0	0	0
4	2	0	0	19	0	0	0	0
5	0	0	2	0	23	2	3	0
6	0	3	1	0	1	57	0	0
7	0	0	0	0	0	3	59	0
8	0	1	1	0	0	0	0	28

Table 2: Matrix showing the predicted concepts (y-axis) against the actual concepts (x-axis) for each record in the training set

case base. Stage 2 (in Figure 1) was now repeated using concepts as the dependent variable instead of clusters. Table 2 shows the results of the knowledge validation in

One tuple will eventually be chosen from each concept to become a case in the

this case. It is clear from Table 2, that the concepts are distinct with no significant interaction between them. Also, the rules discovered using C4.5 in this case provide

descriptions of the individual concepts which serve the dual purpose of classifying new data into the concept that it belongs to as well as providing a semantic understanding of each concept. The model will be used on the target case during problem solving, to identify the concept case base that must be utilised for case retrieval. Figure 3 shows the distribution of the tuples in the original data set for each concept while Figure 4 shows

Value	Proportion	%	Occurrences
1		17.1	151
2		7.7	68
3		3.96	35
4		7.93	70
5		9.29	82
6		19.48	172
7		20.72	183
8		11.21	99
default		2.6	23

Figure 3: Concept distribution in the database
some example rules that were discovered.

The next stage was to test whether each concept contains enough information within it to be able to predict the price of houses within the concept. Clearly, only if this is the case would the concept be worth storing within the case base. This is due

Characteristics of Concept 9:

If Type = 2 and Bedrooms = 3 and Reception = 1
 Or If Price > 40500 and Type = 5 and Reception = 1

Characteristics of Concept 1:

If Type = 1 and Bedrooms = 2 and Reception = 1 and Repair = 2
 Or if Date = "1st Quarter of 93" and Type = 1 and Bedrooms = 2 and Reception = 1
 Or if Price <= 34500 and Age = 1 and Reception = 1 and Repair = 2
 Or if Type = 6 and Bedrooms = 2
 Or if Date = "2nd Quarter of 93" and Type = 1 and Age = 2 and Reception = 1 and Repair = 1
 Or if Date = "2nd Quarter of 93" and Type = 1 and Bedrooms = 4 and Reception = 1 and Repair = 2
 Or if Date = "3rd Quarter of 93" and Type = 1 and Age = 5 and Reception = 1 and Repair = 2

Figure 4: Example rules produced by C4.5 algorithm

to the fact that if enough information were not available from the concept the discovered adaptation knowledge would not be of a high quality resulting in poor competence of the CBR system due to poor *reachability* of the cases. To this end, we

trained a regression tree on each of the individual concepts with Price of the houses as the dependent attribute. Once again the holdout sample technique was used to validate the discovered regression tree. The results are summarised in Table 3 below.

Concept	Mean Absolute Error	Maximum	Minimum	Mean	S.D
1	8502.5	64000	1200	21794.0	12255.0
2	9457.2	70000	2900	19,629.4	16497.7
3	67915.3	400000	39750	114182.9	79489.7
4	5676.4	47000	4550	23647.1	7578.9
5	25673.6	35000	22250	60775.5	24900.7
6	6497.5	68000	2500	22506.1	14037.7
7	7317.2	70000	3000	34330.9	11988.2
8	7566.0	93000	1500	32757.1	13014.4

Table 3: Results of cluster analysis

As can be seen in Table 3, two of the concepts, namely concepts 3 and 5, have mean absolute errors much greater than the other concepts. Also, worth noting is the fact that concepts 3 and 5 have standard deviations and mean values for price greater than the other concepts. These records can be dealt with in two possible ways. They can either be refined to reduce the mean absolute error or each of the records in these concepts can be added as a case to the case base as individual cases. These outliers would correspond to the Auxiliary cases within the case base [SMYT95]. The former option was chosen in the case of concept 5 and the latter for concept 3. Tuples in concept 5 were split into two subsets using the price threshold value of £93,000 (the maximum price of a house in the concepts with acceptable mean absolute error). The mean absolute error for the subset with price less than £93,000 was £8670.8 and was defined as the new concept 5. The tuples in concept 5 with price greater than £93,000 were taken to be outliers.

A model was built around the raw data set to predict the price of houses without any of the clustering techniques described above. The results of this provided a baseline to judge the effectiveness of the technique. The mean absolute error on analysis of this model was approximately £14000. It can be seen from Table 1 that utilising the clustering techniques described above, reduced this mean absolute error value.

We are now in a position to choose the individual cases for the case base. Of the overall 883 records present in the data set initially which had the potential to become cases the number of cases actually identified using this method was reduced to only 75 - a reduction in number of potential cases of 1:11. The identified cases consist of the 23 outliers identified by the Kohonen network in the first phase of the process (shown as 'default' in Figure 3), 35 tuples of concept 3, 10 outliers identified in the original concept 5 and 1 record chosen at random from each of the concepts 1, 2, 4, 5, 6, 7 and 8.

In the next section we discuss the technique used to discover adaptation knowledge from each of these concepts. The adaptation knowledge discovered in this

way would result in the concept defining the *reachability* of the chosen case. The adaptation knowledge from each of the concepts may then be used to adapt the case chosen from the cluster whenever it is the most similar case to the target.

4 Discovery of Adaptation Knowledge

The Adaptation Knowledge base consists of knowledge discovered from the Enhanced Operational Database. The knowledge representation used is rule based and has the following format:

- **Reference Concept:** This contains an identifier for the concept that the rule has been discovered for.
- **Context:** This is a conjunction of expressions defined on attributes within the case base. Each expression may contain internal disjunctions. This contextual expression must be satisfied for the rule to apply to the retrieved case.
- **Antecedent:** The antecedent contains the attribute differences that can be adapted by using the rule. It consists of a conjunction of simple expressions.
- **Consequent:** The consequent contains the change that occurs in the dependent attribute of the case base when the adaptation represented by the rule is applied.
- **Confidence:** This is a measure of belief in the accuracy of the rule. The definition of confidence of the rule depends on the rule discovery algorithm used by the Data Mining Engine as well as on the type of the dependent attribute.

Example: A rule that may be discovered from a housing database may be:

Reference Concept: C_i

Context: (Type = Semi-detached) and num_beds = [1, 4]

Antecedent: (num_beds increases by 1) and (CoveredArea increases by 10 sq. m)

Consequent: +4000

Confidence: 98%

This rule essentially reads as: "For semi-detached houses that have number of bedrooms ranging from 1 to 4, an increase in number of bedrooms by 1 room and a increase in covered area by 10 square metres results in the price of the house increasing by 4000 pounds". This rule has a 98% confidence associated with it.

The Data Mining Engine within the M² system consists of the Mining Kernel System [ANAN97] (MKS) implemented at the authors' laboratory. The type of adaptation knowledge discovered by the system is dependent on the format of the cases in the case base as well as the retrieval algorithm within the CBR Engine.

The types of domain knowledge suggested by [HANN96] that need to be utilised by the Data Mining algorithms are taken into account by various components of the M² system. Totally irrelevant features are not stored as part of the case base. The Case Constructor component ensures that these features are not present in the case base. Contextually relevant features are taken care of by the CBR Engine component as we consider this to be domain knowledge that must be utilised by the similarity measure defined on the cases in the case base, rather than an aspect of the adaptation process. Contextual knowledge as well as feature interactions must be derived from the data during the discovery of general adaptation rules. As for known adaptation knowledge, we take a softer approach than that utilised by Hanney et al.

We assume that this knowledge is to be verified before use. If this knowledge is not consistent with the data stored in the Enhanced Operational Database operational tables, this should be brought to the notice of the domain expert who must either retract the rule or must deal with the noisy data within the database that is causing the rules to be inconsistent with it.

The Data Mining Engine within the M² system uses rule induction techniques to discover the required adaptation rules. The general induction process consists of two parts:

- Search Space Partition Rules Generation: The search space partition rules are regularities in the data that associate attributes in the data with the outcome field. The induction process utilises generalisation and specialisation heuristics to induce general patterns without over generalising or specialising them. The induction heuristics used here are the *extending reference rule*, *climbing generalisation tree rule* and *closing interval rule* for generalisation and *adding condition rule* for specialisation [MICH83]. Search space partition rules are discovered using a variant of the Association rule discovery algorithms suggested in literature [AGRA94].
- Adaptation Rule Generation: During this phase, the search space partition rules discovered are further generalised using the *dropping condition rule* [MICH83] and combined to produce rules that associate changes in the data attributes to changes in the outcome field, producing context expressions in the process.

The algorithm within MKS utilised for discovery of adaptation knowledge is a variation of the association rule discovery algorithm. The syntactic constraint, 'attribute Price as consequent', is used to constrain the rules discovered to the required format. The criterion used for stopping specialisation of rules discovered is based on the co-efficient of variation of Price defined as the standard deviation*100/ mean. Whenever, the co-efficient of variation is less than a user defined threshold, specialisation stops. Also, each specialisation of a rule discovered in the previous iteration of the algorithm has its context automatically defined as the antecedent of the rule being specialised.

5 Concluding Remarks and Future Work

In this paper we discussed the process within the M² CBR system for acquiring case knowledge in the form of cases and adaptation knowledge using Data Mining techniques for clustering, classification and association rule discovery. In particular we discussed the use of Kohonen Networks for discovering an initial clustering of the data tuples, the use of C4.5 to analyse the efficacy of the clustering and regression tree induction to assess the information content of the developed concepts. We then discussed the approach to choosing cases from the concepts. This process obtained a reduction in size of the case base from 883 to 75 cases with each case either being a pivotal or an auxiliary case within the case base.

The results presented here are clearly promising, however, a number of issues within the M² system have not been utilised as yet. We now discuss some of the issues we hope to address in the near future. Rough set analysis [PAWL95] has

recently gained a lot of attention in the field of Data Mining. One of the major uses of rough set analysis is to filter out irrelevant attributes by discovering *reducts*. This technique may be used for each concept discovered to remove irrelevant attributes with respect to the concept before case construction.

Further developments within the M² system will utilise Data Mining for refining the cases and adaptation knowledge improving competence of the case base as well as utilising negative cases. Also techniques that would reduce the dependence of output knowledge on user parameters will be investigated.

6 References

- [AGRA94] R. Agrawal, R. Srikant. Fast Algorithms for Mining Association Rules in Large Databases, *Proc. of the 20th Int. Conf. on VLDB*, pp. 487 - 499, Chile, 1994.
- [ANAN97] S. S. Anand, B. W. Scotney, M. G. Tan, S. I. McClean, D. A. Bell, J. G. Hughes, I. C. Magill. Designing a Kernel for Data Mining, *IEEE Expert*, pp. 65 - 74, April, 1997.
- [ANAN97a] S. S. Anand, W. Dubitzky, D. Patterson, A. Schuster, J. G. Hughes. M²: A First Step Towards Automated Generation and Updating of Case-Knowledge from Databases, Internal Report, Faculty of Informatics, University of Ulster, 1997 (available from <http://iserve1.infj.ulst.ac.uk:8080/m2.ps>).
- [ASHL88] K.D. Ashley, E.L. Rissland. A case-based approach to modelling legal expertise, in *IEEE Expert*, 3(3), pp. 70-77, 1988.
- [CLEM96] CLEMENTINE User Guide, Integral Solutions Ltd, Basingstoke, England, 1996.
- [CURE96] O. Curet, J. Elliott, M. Jackson. Designing knowledge discovery based systems in business, finance and accounting with a case-based approach: two case studies, IEE Colloquium on Knowledge Discovery and Data Mining, 1996.
- [DUBI96] W. Dubitzky, J. G. Hughes, D. A. Bell. A Generic, Object-Oriented Case-Knowledge Representation Scheme, and its Integration into a Wider Information Management Scenario, in *Expert Systems: The International Journal of Knowledge Engineering and Neural Networks*, vol. 13 (3), pp. 219-233, Blackwell Publishers, UK, 1996.
- [DUBI96a] W. Dubitzky, J.G. Hughes, D.A. Bell. Case Memory and the Behaviouristic Model of Concepts, in *Proc. Advances in Case-Based Reasoning, 3rd European Workshop, EWCBR-96*, pp 120-134, Switzerland, 1996.
- [DUBI97] W. Dubitzky, A. Schuster, J.G. Hughes, D.A. Bell, K. Adamson., How Similar is VERY YOUNG to 43 Years of Age? On the Representation and Comparison of Polymorphic Properties, *15th International Joint Conference on Artificial Intelligence*, Japan, 1997.
- [FAYY96] U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, R. Uthurusamy (Editors). *Advances in Knowledge Discovery and Data Mining*, AAAI/ MIT Press, 1996.
- [GEBH94] F. Gebhardt. *Discovering interesting statements from a database*, Applied Stochastic Models and Data Analysis, Vol. 10, pp. 1-14, 1994.
- [HANN96] K. Hanney, M. T. Keane. Learning Adaptation Rules From a Case-Base, in *Proc. of European Workshop on Case Based Reasoning*, pp. 179 - 192, 1996.
- [MICH83] R. S. Michalski. A Theory and Methodology of Inductive Learning, in *Machine Learning: An Artificial Intelligence Approach* ed. R. S. Michalski, J. G. Carbonell, T. M. Mitchell, pp. 83 - 134, 1983.
- [PAWL95] Z. Pawlak, J. Grzymala-Busse, R. Slowinski, and W. Ziarko. Rough Sets, *Communications of the ACM*, Vol. 38, pp. 89—95, 1995.
- [SHUS97] A. Schuster, W. Dubitzky, D.A. Bell, J.G. Hughes, K. Adamson. Aggregating Features and Matching Cases on Vague Linguistic Expressions, *15th International Joint Conference on Artificial Intelligence*, Japan, 1997.
- [SMIT81] E. E. Smith, D. L. Medin, *Categories and Concepts*, Harvard University Press, Cambridge, Massachusetts, 1981.
- [SMYT95] B. Smyth, M. T. Keane. Remembering to Forget: A Competence-Preserving Case Deletion Policy for Case-Based Reasoning Systems, in *Proc. of IJCAI-95*, pp 337 - 382, 1995.

Discovery of Association Rules over Ordinal Data: A New and Faster Algorithm and Its Application to Basket Analysis

Oliver Büchter and Rüdiger Wirth

Daimler-Benz Research & Technology FT3/KL
PO BOX 2360
89013 Ulm, Germany

oliver.buechter@dbag.ulm.daimlerbenz.com

Abstract

This paper argues that quantitative information like prices, amounts bought, and time can give valuable insights into consumer behavior. While Boolean association rules discard any quantitative information, existing algorithms for quantitative association rules can hardly be used for basket analysis. They either lack performance, are restricted to the two-dimensional case, or make questionable assumptions about the data. We propose a new and faster algorithm Q2 for the discovery of multi-dimensional association rules over ordinal data, which is based on ideas presented in [SA96]. Our new algorithm Q2 does not search for quantitative association rules from the very beginning. Instead Q2 prunes out a lot of candidates by first computing the frequent Boolean itemsets. After that, the frequent quantitative itemsets are found in a single pass over the data.

In addition, a new absolute measure for the interestingness of quantitative association rules is introduced. It is based on the view that quantitative association rules have to be interpreted on the background of their Boolean generalizations.

We experimentally compare the new algorithm against the previous approach, obtaining performance improvements of more than an order of magnitude on supermarket data. A rather astonishing result of this paper is that an additional run through the transactions does pay off when searching for quantitative association rules.

Keywords

association rules, basket analysis, quantitative association rules, ordinal data

1 Introduction

1.1 Background

The discovery of association rules has attracted a lot of attention in the field of data mining and knowledge discovery since it was first described in 1993 [AIS93]. It addresses the problem of finding *all* the rules that describe associations between the items within a *large* transaction data base and that fulfill certain conditions. Rules

over relations are of the form $C_1 \Rightarrow C_2$ where C_1 and C_2 are conditions on tuples of the relation. Most often, the conditions are restricted to be simple equality predicates restricting attribute values or conjunctions of such predicates on different attributes. Conditions for the rules often express lower limits for the basic evaluation measures support and confidence.

The discovery of association rules has been motivated by market basket analysis. Progress in bar-code technology lets supermarkets store what items are purchased in every transaction. Business decisions take advantage of knowledge about which items are frequently purchased together or in other words: which items are associated. So the problem of finding associations is defined as finding sets of items that are frequently purchased together. The fraction of transactions containing them together among all the transactions is called the support of the itemset. In order to express associations as rules, the itemset is divided into two sets C_1 and C_2 and the confidence is computed, how many of the transactions containing C_1 are also containing C_2 .

The term association rule discovery now refers to a whole family of particularly efficient data mining methods which are (quasi) linear in the number of transactions. A lot of new methods have been developed which not only improve performance but which sometimes also vary basic principles. So it is getting hard to define common ground and differences. We will later fit our results into the still growing stream of current publications.

1.2 Quantitative Association Rules for Basket Analysis

The domains of the attributes have originally been restricted to Boolean domains [AIS93][AS94]. The introduction of quantitative association rules in [SA96] has generalized this definition to arbitrary domains but especially focused on the use of quantitative information. Conditions concerning quantitative attributes are expressed as membership in subintervals. While [SA96] introduced the idea of quantitative association rules, the application to basket analysis has not been addressed. Instead data with only 10 items have been chosen for experiments.

But supermarket data contain quantitative information that can give valuable insight into consumer behavior: the time of the transaction, the amounts bought, and the prices of the items. So an efficient quantitative association algorithm can find price ranges where an association is particularly strong, can build more homogeneous customer subgroups, and can automatically detect time intervals, where special associations occur (because different times attract different customers). This may lead to automatic detection of the often-quoted association between beer and diapers [Bel97]:¹

¹ Unfortunately, there is no reliable reference for this discovery. Instead, we have heard that this discovery is completely fictitious and that until now no supermarket has been found where an association between beer and diapers may be observed. In our data, there is definitely no association between beer and diapers.

„The classical example is the observation that, in the evenings, beer and diapers are often purchased in the same transaction. Presumably this is not because babies like a glass of beer before they go to bed, but because the father is sent out to buy diapers when they run out and chooses to purchase beer at the same time. This may lead to store layouts which (say) ensure that anyone buying diapers must pass through the beer section.“

A classical association rule method would have to rely on a manual coding of every possible time interval into a Boolean attribute in order to find the characterization in the evening. This requires the right hypothesis from the data miner. For a quantitative association rule algorithm coding the time into a numeric attribute is sufficient.

The general potential of quantitative association rule algorithms for market basket analysis is the possibility of flexibly characterizing homogeneous groups of customers not only by ‘what they buy’ but also by amounts, prices, and the time. In comparison, Boolean or hierarchical approaches have to rely on manual preparation of the data according to existing hypotheses. So an efficient quantitative association rule algorithm broadens the explorative potential of association rule discovery.

In this paper, we first review existing algorithms for discovering quantitative association rules and examine the requirements from market basket analysis. According to these requirements we propose a new and more efficient algorithm for the discovery of quantitative association rules over ordinal data. We experimentally compare the new algorithm against the previous approach on supermarket data, obtaining improvements of more than an order of magnitude. In addition, we define the information content of a quantitative association rule and derive an absolute interestingness measure for quantitative rules. An absolute measure may be computed in constant time, while the relative interestingness measure from [SA96] needs considerable computation.

2 Related Work

Classical Boolean association rules [AIS93][AS94] can only express whether an item is contained within a transaction or not. Numerical Information concerning the items can be used for specializing items into subclasses (like ‘cheap’, ‘normal’, ‘expensive’). This is done in order to find subclasses as homogeneous as possible, so that an association becomes as strong as possible (high confidence). Autonomous construction of subclasses imposes the problem of trying combinations of possible discretizations for the numeric attributes. This is a reason why the quantitative association rules problem is harder than the Boolean association rules problem. Different strategies for handling the problem have been applied:

1. [FMM96] and [YFM97] have restricted themselves to the problem of finding *two-dimensional* association rules between numeric attributes. This problem can be mapped to the problem of finding regions of pixels. While these methods may be used to find a detailed characterization where the association between *two* items is most convincing, they cannot help in finding out multivariate associations that have to be characterized by more than two numeric attributes.

2. [SA96] have introduced not only the quantitative association rules problem but also an algorithm QAR² based on the Apriori-algorithm [AS94]. QAR finds multi-dimensional quantitative association rules like the following: if shampoo with a price between four and six dollars is bought in the time between two and eight o'clock, then in 50 % of the cases also a rinse is bought for a price between five and seven dollars. In this kind of rule, subintervals for the three items 'shampoo', 'time', and 'rinse' are specified by the algorithm. The goal of multi-dimensional quantitative association rules is finding subintervals where the items are stronger (or weaker) associated than in general.

There is a serious complexity problem in trying out every possible subinterval against combinations of all others. In QAR, this problem is cut down by discretizing all quantitative attributes into intervals, which are called *buckets*, using equi-depth-partitioning. This means that every bucket of an item will contain nearly the same number of transactions.³ These buckets are used as items afterwards, when the algorithm counts „bucket set“ occurrences rather than item occurrences. But finally neighboring „bucket sets“ are combined in order to build compound interval-combinations. Note that the resulting interval-combinations are required to have minimum support while the intermediate bucket-combinations are not. So candidate pruning is a little bit more difficult than in Apriori: a bucket-combination is a candidate if the Boolean generalization would be a candidate in Apriori. The algorithm has a major performance problem: Suppose, every item is discretized into b buckets and we have to count candidates of size i (containing i items), then for every candidate Apriori would generate, QAR generates b^i buckets.

3. MAQA [ZLZ97] partitions every numeric attribute separately. Clustering finds regions that are separated by local minima of the frequency distribution. These regions of an attribute are used as items. Finally, when rules are found that contain the same conditions except for one condition concerning the same attribute and describing neighboring regions, rules are combined to form a new (more general) one.
4. [MY97] propose a method for discovering association rules over interval data called „distance-based association rules“ (DAR). Their approach is heavily inspired by cluster analysis. Values of interval data are not only ordered, even the separation between data points has meaning. This is an important condition for clustering, because it relies on distance measures.

For the rest of this section, we would like to discuss which assumptions can be made about quantitative supermarket data. These assumptions effect the choice of an appropriate algorithm. We want to argue in favor of the QAR approach, because it does

² [SA96] used no name for their algorithm but [ZLZ97] already called it QAR.

³ QAR differs from other association rule algorithms in counting zeros just like every other number. In basket analysis, describing what is *not* contained in 99 % of the baskets is a waste of time (and makes mining computationally intractable). So if we speak of QAR in the rest of this paper, we mean a QAR that does not count zeros.

without (1) the assumption of interval scale and (2) the definition of an appropriate similarity measure.

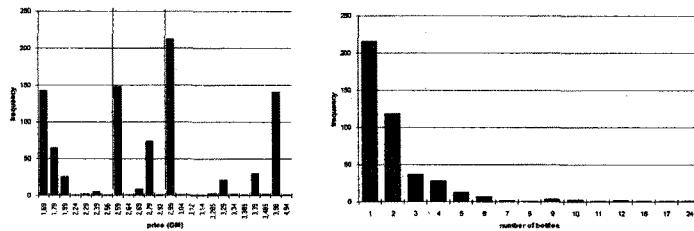


Figure 1: Frequency distributions of quantitative supermarket data

Though there is a lot of quantitative information like amounts, prices, and time we should *not* suppose that it is interval data. The same distance between two data points has different meaning depending from its location within the attribute's range. For instance, the buying decision for a cheap product may be made because of ten cents - the price plays a major role here. More expensive articles are bought because of certain qualities, a better design, taste, or more political correctness. Then a price difference of several dollars does not matter. Figure 1 gives two examples of frequency distributions from market basket analysis. Prices usually are not distributed equally but crowd on certain critical points. This results in a lot of local maxima of the frequency distribution. So MAQA would choose a very fine partitioning. On the other hand the amount bought has no extremum in the relevant range but two chance maxima at 9 and 12. So MAQA would be unable to find rules like „who buys a lot of white wine buys also a lot of red wine“. So whether a quantitative information can be supposed to be of interval quality or only ordinal depends form the sample of cases and from knowledge about the target objects. As association rule discovery is used as an exploratory method it is safer to do without the assumption of interval quality. To cut a long story short, if pure distance-based partitioning is applied to supermarket data there is a strong bias imposed by the meaningless (!) intervals, because quantitative information in supermarket data is only ordinal data.

There are also clustering approaches for non-interval data, but they rely on the definition of an appropriate similarity measure over the heterogeneous dimensions of quantitative data. From our point of view, mining of quantitative association rules can be applied best for exploratory data analysis the fewer assumptions about the scale and about interrelations between different dimensions have to be made. In other words, quantitative association rule methods should be applied to find out about the interrelation between dimensions, not by defining these interrelations into similarity measures beforehand.

If we need the exploratory power of more-than-two-dimensional association rules, QAR is a conceptually good but not very efficient choice for exploring quantitative dimensions of supermarket data.

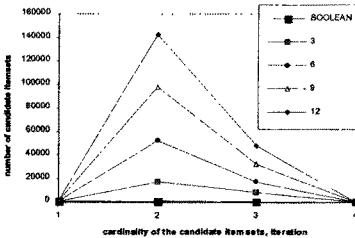


Figure 2: Number of candidates in different iterations under variation of the discretization granularity

3 Performance Analysis of QAR

Figure 2 shows the number of candidates generated in every iteration for a number of test runs of QAR over supermarket data. It is proportional to the number of buckets per item that has been varied over the test runs. Buckets are the partitions of the range of an item, which are counted. These results have two major implications:

- The number of buckets should be chosen carefully by the user, in order to avoid hours or days of computation during an interactive session.
- To avoid a combinatorial explosion of the number of buckets only those buckets should be counted that are really needed. This is a general idea of improvements to association rule algorithms: generate only few candidates that will not be frequent.

The rest of this paper is motivated by the following hypothesis:

Hypothesis 1: The major part of the bucket combinations that QAR counts does not belong to any frequent itemset.

Table 1: Candidate bucket combinations versus frequent itemsets

iteration	number of candidate bucket combinations	number of frequent Boolean itemsets	upper bound for the number of necessary bucket combinations
1	670	64	192
2	3468	79	711
3	3129	10	270
4	1209	0	0

In order to verify this hypothesis we would like to start with a little experiment: Supermarket data are analyzed with QAR. These contain 670 quantitative items and 20000 transactions. For this little experiment, we choose a minimum support of 5%

and a discretization into $d=3$ intervals for each item.⁴ Table 1 shows the results of this experiment. The second column contains the number of candidate bucket combinations, while the third column shows the number of frequent Boolean itemsets. The last column shows an upper bound for the number of necessary bucket combinations, which is computed from the number of frequent Boolean itemsets according to the consideration that a k -itemset is counted by up to d^k buckets. In this example QAR works with 8476 buckets, but only 1173 of them belong to frequent itemsets. Only 14 percent of the buckets were really necessary.

4 Q2

4.1 Basic Idea

The basic idea of our new algorithm Q2 is to count only those combinations of buckets, that are contained in the interval combinations of frequent itemsets. In other words: the overhead of counting lots of buckets is reduced towards counting only buckets for successful candidates.

Because of the monotonicity of the support measure [Toi96] specializing an itemset by introducing an additional item or by constraining the range of a quantitative item can never lead to a higher support. Consequently, all frequent quantitative itemsets are specializations of frequent quantitative itemsets with unconstrained ranges. These are extensionally the same as the Boolean itemsets that (for example) Apriori finds.

Therefore, the set of bucket combinations BK that Q2 needs can be determined from the set of Boolean itemsets.. Let B be the set of all buckets, L the set of frequent Boolean itemsets and the predicate $\text{subinterval}(b_i, l_i)$ be true, iff bucket $b_i \in B$ counts a subinterval of the frequent itemset $l_i \in L$.

$$BK := \bigcup_{k=1..n} \left\{ \{b_1, \dots, b_k\} \subset B^* \mid \exists \{l_1, \dots, l_k\} \subseteq L, l_i \in L : \forall i \in \{1, \dots, k\} : \text{subinterval}(b_i, l_i) \right\}$$

Q2 counts just the bucket combinations in BK , whose buckets all count a subinterval of the range of a frequent Boolean itemset. Because the frequent Boolean itemsets have to be determined first, Q2 is a performance improvement over QAR if the following hypothesis can be proved:

Hypothesis 2: An additional run through the transactions in order to count the bucket combinations is more than compensated by the reduction of bookkeeping effort. So Q2 can be faster than QAR and uses less memory.

In order to test this hypothesis we have implemented QAR and Q2 (cp. following section) and later carried out evaluation experiments (cp. section 5). These experiments show that Q2 is more than a magnitude faster than QAR on supermarket data that we regard as interesting for quantitative analysis.

⁴ For this argument, the concrete choice of d does not matter. The choice of the minimum support may together with statistical particularities of the data influence the fraction of successful candidates, but there is no systematic influence.

4.2 Implementation of Q2

Q2 uses a prefix tree to store quantitative itemsets of different cardinality, so that they may be counted in a single pass over the transactions. A prefix tree stores counters for itemsets at internal and external nodes. Which itemset is counted is defined by the path to the node. A prefix tree is a little bit faster (but needs more memory) than a hash tree (as used in [SA96]) if no hashing is done. Figure 3 shows the main algorithm of Q2. First all frequent Boolean itemsets (with at least two elements) are found for example with Apriori. Now only the items contained in these sets are discretized according to the specification by the user (this is a second, but minor performance improvement over QAR). A method Q2-gen now generates the prefix tree with just the bucket combinations BK that have to be counted for the discretized items. Then the prefix tree is used to count these bucket combinations within a single pass over the data. Finally, all R-interesting rules are generated from the prefix tree (like in QAR, but again the tree is smaller and so a third, minor performance improvement is reached).

```

(1) L = apriori( Inputfile );
(2) Items = { i ∈ I | ∃ l ∈ L: i ∈ l };
(3) Data.read_and_discretize(Inputfile,SymbolTable,Items );
(4) Prefixtree.Q2gen( L );
(5) forall ( t ∈ Data )
    Prefixtree.count_all_subsets_of( t );
(6) Prefixtree.generate_quantitative_rules;

```

Figure 3: Algorithm Q2

Apart from this general idea, an efficient solution for another problem makes up the performance of Q2: The generation of all the candidate bucket combinations e.g. the construction of the whole prefix tree in a single step is a critical factor for the efficiency of Q2. The knowledge of which Boolean itemsets are frequent must prevent us from transferring the whole candidate generation effort of QAR into Q2. So candidate generation in Q2 is done by working off the list of frequent itemsets: for each of them all „their“ buckets are inserted into a new prefix tree.

4.3 Selecting interesting quantitative rules

The use of support and confidence as evaluation criteria for association rules has been criticized by various authors. For quantitative association rules the evaluation of the confidence makes even less sense. Suppose 2% of all the transactions contain shampoo and the following rule has been found: 50% of the people that buy a rinsing also buy a shampoo. Now a quantitative association rule could further characterize *which* shampoo is bought or *how much*. But this would specialize the consequence of the rule, so that the support of „rinsing and specialized shampoo“ could only shrink which lowers the confidence. So a fact like „expensive shampoo is almost always

bought together with rinsing“ will be filtered out, since its confidence will be below 50 %. A far better selection measure is the dependence which is computed according to the following formula [BMS97]:

$$\text{dependence}(A, B) := \frac{P[A \wedge B]}{P[A]P[B]}.$$

The measure expresses the deviation of the frequency of co-occurrence of A and B from the expectation that is based on the assumption of no association between A and B.

In analogy to this measure we can define the information content of a quantitative rule: a quantitative rule is only interesting, if the observed co-occurrence for the special ranges deviates from the expectation that is based on the corresponding Boolean association rule. Let $bg(X)$ be the corresponding Boolean itemset to the quantitative itemset X . The degree of deviation is expressed by the following measure q-dependence:

$$q\text{-dependence}(A, B) := \frac{P[A \wedge B]}{P[bg(A) \wedge bg(B)] \frac{P[A]}{P[bg(A)]} \frac{P[B]}{P[bg(B)]}}.$$

The denominator computes the expectation for co-occurrence of the itemsets A and B . The corresponding Boolean rule tells about the co-occurrence of the Boolean generalizations $bg(A)$ and $bg(B)$. The result $P(bg(A) \wedge bg(B))$ is multiplied by the relative size of the quantitative itemsets with regard to their Boolean generalizations. The numerator shows the observed co-occurrence of A and B .

5 Evaluation

The performance of QAR and Q2 can only be evaluated on real supermarket data.⁵ The following three experiments use 25000 transactions from a German supermarket on the level of article groups (like ‘beer’ and ‘diapers’). The data contain 600 of these groups but the analysis is restricted to 69 of them which are associated in a great number of the baskets of goods. This restriction corresponds to the pragmatic procedure of a data miner to focus on important details. Without these restrictions QAR could not be used at all.

The selection of the parameter *minimum support* is a main factor for the execution time of every association rule algorithm. The definition of the minimum support determines which items and itemsets are considered frequent. If this parameter is lowered itemsets with more and more elements are found. In order to find an itemset with n elements level-wise algorithms like Apriori and QAR need n runs through the transactions. So the number of runs through the transactions and the complexity of every run grow simultaneously.

⁵ For Boolean and hierarchical association rules there is a dataset generation program freely available over the internet, but for quantitative data there is none yet.

Quantitative association rule algorithms (QAR, MAQA, Q2) count quantitative itemsets using a lot of bucket combinations. Their number grows exponentially with the cardinality k of the itemsets. Consequently the complexities of counting and of candidate generation grow exponentially with the number of iterations.

Figure 4 shows the execution times of QAR and Q2 under variation of the minimum support parameter in a series of experiments with supermarket data. The discretization has been fixed to six buckets for each item, the confidence has been set to 60 percent. For a minimum support lower than three percent (supports greater than this are not so fruitful for basket analysis) the execution time of Q2 is more than a magnitude lower than that of QAR. In other words: Q2 can find rules with half the support during the time that QAR needs (the vertical lines show the transition to 4- and 5-itemsets).

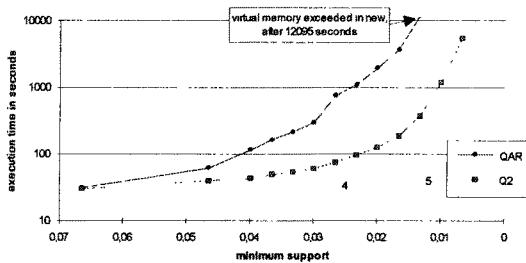


Figure 4: Performance comparison under variation of the minimum support

The *number of buckets per item* has exponential influence on the execution times, too. But this effect is by far weaker for Q2 than for QAR. Figure 5 shows the results of a few experiments varying the number of buckets.

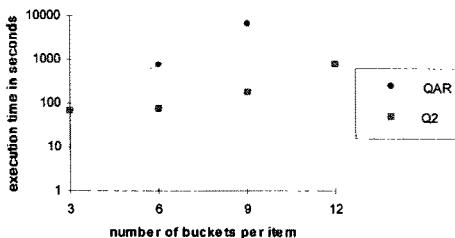


Figure 5: Performance comparison under variation of the number of buckets

The share of counting in the whole execution time is rising with the number of transactions. It could be argued that the additional pass over the data is also getting more and more costly, but this rise is still proportional to the rise for the rest of the counting effort. Consequently, the advantage of Q2 should nearly be independent from the number of transactions. Figure 6 compares the scalability of Q2 and QAR. The x-axis shows the size of the data used. The y-axis contains the execution time used by the

algorithm. As expected, the execution time of QAR rises faster than Q2's execution time.

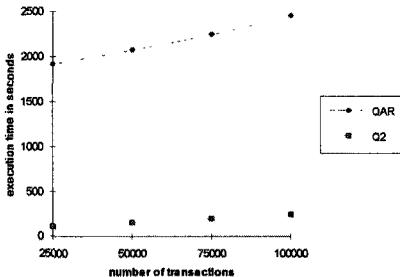


Figure 6: Performance comparison under variation of the number of transactions

Recently, [ZPO97] have proposed a new algorithm for the discovery of Boolean association rules which benefits from vertical data and beats Apriori. Future work will show if this approach can be transferred to quantitative association rules and further speed up the mining of quantitative association rules. Anyway, Q2 can already take profit from this work since Apriori can simply be replaced by every algorithm that finds Boolean association rules.

6 Conclusion

The new algorithm Q2 for the discovery of quantitative multi-dimensional association rules over ordinal data was presented. Experiments with supermarket data showed that it performs much better than the existing algorithm QAR. Apparently it is better to search for Boolean association rules first and to search for quantitative rules afterwards in one additional run through the transactions than to do expensive bookkeeping of mostly unsuccessful counters.

In addition, ideas have been presented how basket analysis can take advantage of quantitative information with the aid of our algorithm. Because Q2 is less sensitive to variation of the basic parameters minimum support and number of buckets per item, it can much better be used in interactive session than QAR, when the user does not want (or is not able) to judge the complexity of his data mining queries.

A rather astonishing result of this paper is that an additional run through the transactions does pay off for the search for quantitative association rules, while for Boolean association rules the faster algorithms do as few runs as possible and do without informed pruning [ZPO97].

References

- [AIS93] Rakesh Agrawal, Tomasz Imielinski, and Arun Swami. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD Conference, Washington DC, USA, May 1993*, 1993.
- [AS94] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules. In *Proceedings of the 20th VLDB Conference, Santiago, Chile*, 1994.
- [Bel97] Bell, Michael: *A Data Mining FAQ*, 1997, [[Http://www.qwhy.com/dmfaq.htm](http://www.qwhy.com/dmfaq.htm), 23.07.97].
- [BMS97] Sergey Brin, Rajeev Motwani, and Craig Silverstein. Beyond market baskets: Generalizing association rules to correlations. In *1997 ACM SIGMOD Conference on Management of Data*, pages 265--276, 1997.
- [FMM96] Takeshi Fukuda, Yasuhiko Morimoto, Shinichi Morishita, and Takeshi Tokuyama. Mining optimized association rules for numeric attributes. In *Proceedings of the 15th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*. Montreal, Canada, 1996.
- [MY97] R.J. Miller and Yang Y. Association rules over interval data. In *ACM SIGMOD 1997, Tucson, Arizona*, May 1997.
- [SA96] Ramakrishnan Srikant and Rakesh Agrawal. Mining quantitative association rules in large relational tables. In *Proc. of the ACM SIGMOD Conference on Management of Data, Montreal, Canada*, June 1996.
- [Toi96] Hannu Toivonen. *Discovery of Frequent Patterns in Large Data Collections*. PhD thesis, University of Helsinki, Department of Computer Science, November 1996.
- [YFM97] Kunikazu Yoda, Takeshi Fukuda, Yasuhiko Morimoto, Shinichi Morishita, and Takeshi Tokuyama. Computing optimized rectilinear regions for association rules. In *Proceedings of the 3rd International Conference on KDD and Data Mining*, Newport Beach California, August 1997.
- [ZPO97] M.J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li. New algorithms for fast discovery of association rules. In *Proceedings of the 3rd International Conference on KDD and Data Mining*, Newport Beach California, August 1997.
- [ZLZ97] Zhaojun Zhang, Yuchang Lu, and Bo Zhang. An effective partitioning-combining algorithm for discovering quantitative association rules. In *PAKDD97*, 1997.

Effect of Data Skewness in Parallel Mining of Association Rules

David W. Cheung and Yongqiao Xiao

Department of Computer Science
The University of Hong Kong, Hong Kong
Email: {dcheung, yqxiao}@cs.hku.hk

Abstract. An efficient parallel algorithm FPM(Fast Parallel Mining) for mining association rules on a shared-nothing parallel system has been proposed. It adopts the count distribution approach and has incorporated two powerful candidate pruning techniques, i.e., distributed pruning and global pruning. It has a simple communication scheme which performs only one round of message exchange in each iteration. We found that the two pruning techniques are very sensitive to data skewness, which describes the degree of non-uniformity of the itemset distribution among the database partitions. Distributed pruning is very effective when data skewness is high. Global pruning is more effective than distributed pruning even for the mild data skewness case. We have implemented the algorithm on an IBM SP2 parallel machine. The performance studies confirm our observation on the relationship between the effectiveness of the two pruning techniques and data skewness. It has also shown that FPM outperforms CD (Count Distribution) consistently, which is a parallel version of the popular Apriori algorithm [2, 3]. Furthermore, FPM has nice parallelism of speedup, scaleup and sizeup.

Keywords: Association Rules, Data Mining, Data Skewness, Parallel Computing

1 Introduction

Mining *association rules* in large databases has attracted a lot of attention in data mining research [1, 2, 4, 6, 9, 13, 15]. The problem can be reduced to finding all *large itemsets* with respect to a given *support threshold* [1, 2]. Two different approaches for this problem have been proposed for parallel systems with distributed memory. One is the *count distribution* approach and the other the *data distribution* approach[3].

In the count distribution approach, each processor is responsible for computing the *local support counts*, which are the support counts in its local partition, of all the candidates. These local support counts are then exchanged among all the processors to compute the *global support counts*, which are the total support counts of the candidates in the database. Subsequently, large itemsets are computed by each processor independently. Only a simple protocol for support count

exchange is required in this approach. Since every processor is required to keep the local support counts for all the candidates at each iteration, one problem of the count distribution approach is the large memory space required to maintain the large number of candidates. Algorithms that use the count distribution approach include CD (Count Distribution) [3] and PDM (Parallel Data Mining) [14].

The data distribution approach attempts to solve the candidate set size problem by distributing the candidates among the processors, i.e., each processor would be responsible for the maintenance of the global support counts of only a subset of the candidates. However, in order to compute the global support counts, the transactions (data) in all partitions must be sent to all other processors. Comparing with the exchange of support counts, sending transactions costs a lot more bandwidth. Algorithms which adopt the data distribution approach include IDD (Intelligent Data Distribution) [8] and HPA (Hash-based Parallel mining of Association rules) [19].

In this work, we propose to investigate parallel mining by adopting the count distribution approach. It requires less bandwidth and has a much simpler communication scheme. To improve the performance of CD and to remedy the possible problem of large number of candidates, we adopt two effective techniques, *distributed pruning* and *global pruning* to reduce the number of candidates at each iteration. These two techniques make use of the information of local support counts of large itemsets found at an iteration to prune candidates for the next iteration. These two pruning techniques have been adopted in a mining algorithm FDM (Fast Distributed Mining) for distributed databases [7]. However, FDM is not ideal for the parallel case, it requires at least two rounds of message exchanges at each iteration, which would not be helpful in order to achieve small response time in parallel computing. We have adopted the two pruning techniques to develop a new parallel mining algorithm FPM (Fast Parallel Mining), which requires only one round of message exchange at each iteration. Its communication scheme is as simple as that in CD. In addition, it can have a much smaller number of candidates due to the incorporation of the pruning techniques.

We have studied the effectiveness of the two pruning techniques. It is shown that they are sensitive to data skewness, which describes the degree of non-uniformity of the distribution of itemsets among database partitions. Both distributed pruning and global pruning are very effective when data skewness is high. We have implemented the algorithm on an IBM SP2 parallel machine. The performance results confirm our observation.

The rest of this paper is organized as follows. Section 2 overviews the parallel mining of association rules. The techniques of distributed and global pruning, together with the FPM algorithm are described in Section 3. In section 4, we define a metric to measure the data skewness of a data partition. In the same section, we have also investigated the relationship between the effectiveness of the pruning techniques and data skewness. Section 5 reports the result of an extensive performance study. Sections 6 concludes the paper.

2 Parallel Mining of Association Rules

Apriori is the most well known serial algorithm for mining association rules [2]. It relies on the `apriori_gen` function to generate the candidate sets at each iteration. CD (Count Distribution)[3] is a parallel version of Apriori. The program fragment of CD at processor i , for the k -th iteration, is outlined in Fig. 1. (X_{sup} , $X_{sup(i)}$ are the global support and locally support at processor i of an itemset X , respectively.) In step 1, every processor computes the same candidate set C_k by applying the `apriori_gen` function on L_{k-1} , which is the set of large itemsets found at the $(k-1)$ -th iteration. In step 2, the local support counts of candidates in C_k are found by scanning the local database partitions. In steps 3 & 4, the local support counts are exchanged with all other processors to get the global support counts and the globally large itemsets L_k are computed independently by each processor. CD repeats steps 1 – 4 until no more candidate is found. An implementation of CD on the IBM SP2 using the MPI (Message Passing Interface) [12] was described in [3].

- 1) $C_k = \text{apriori_gen}(L_{k-1})$;
- 2) scan the local partition to find the local support counts $X_{sup(i)}$ for all $X \in C_k$;
- 3) exchange $\{X_{sup(i)} \mid X \in C_k\}$ with all other processors to get the global support counts X_{sup} , for all $X \in C_k$;
- 4) $L_k = \{X \in C_k \mid X_{sup} \geq \text{minsup} \times |D|\}$

Fig. 1. The CD Algorithm

3 Pruning Techniques and the FPM Algorithm

3.1 Candidate Pruning Techniques

Distributed Pruning The following theorem proved in [7] gives the principle of distributed pruning. Note that the function `apriori_gen` is the same as that in the Apriori algorithm, but it is applied to subsets of L_{k-1} rather than the whole L_{k-1} . Also, we call an itemset X *gl-large* at processor i , if X is both globally large and locally large at processor i .

Theorem 1. *For $k > 1$, the set of all globally large k -itemsets L_k is a subset of $C_k = \cup_{i=1}^n \text{apriori_gen}(GL_{k-1(i)})$, where $GL_{k-1(i)}$ is the set of the gl-large itemsets at processor i .*

Based on Theorem 1, we can prune away a size- k candidate if its size- $(k-1)$ subsets are not all gl-large together at any processor. This pruning technique is called *distributed pruning*. It is straight forward to see that any itemset which can be pruned away by `apriori_gen` will be pruned away by distributed pruning.

Global Pruning In performing distributed pruning at the k -th iteration, the information required for a large itemset is a 1-bit mark for each processor , which indicates whether the large itemset is gl-large at the processor. As a result of the count exchange, the local support counts at each processor of all large itemsets are also available. With this information, another powerful pruning technique called *global pruning* can be developed.

Let X be a candidate k -itemset. At each processor i , $X_{sup(i)} \leq Y_{sup(i)}$, if $Y \subset X$. Therefore $X_{sup(i)}$ is bounded by the value $\min\{Y_{sup(i)} \mid Y \subset X, \text{ and } |Y| = k - 1\}$. Hence the value $X_{maxsup} = \sum_{i=1}^n X_{maxsup(i)}$, where $X_{maxsup(i)} = \min\{Y_{sup(i)} \mid Y \subset X, \text{ and } |Y| = k - 1\}$, is an upper bound of the global support of X . If $X_{maxup} < \text{minsup} \times |D|$, then X can be pruned away. Note that global pruning requires no additional information except the local support counts resulting from count exchange at the previous iteration.

Theorem 2. *If X is a k -itemset ($k > 1$) which can be pruned away at the k -th iteration by distributed pruning, then X can also be pruned away by global pruning.*

Proof: If X can be pruned away by distributed pruning, then there does not exist a processor at which all size $(k - 1)$ subsets of X are gl-large. Thus, at each processor i , there exists a size $(k - 1)$ subset Y of X such that $Y_{sup(i)} < \text{minsup} \times (|D_i|)$. Hence $X_{maxsup(i)} = \min\{Y_{sup(i)} \mid Y \subset X, |Y| = k - 1\} < \text{minsup} \times (|D_i|)$, and $X_{maxup} = \sum_{i=1}^n X_{maxup(i)} < \sum_{i=1}^n (\text{minsup} \times |D_i|) = \text{minsup} \times |D|$. Therefore, X can be pruned away by global pruning. \square

3.2 The Fast Parallel Mining Algorithm (FPM)

We adopt both the distributed and global pruning to develop a new parallel mining algorithm FPM(Fast Parallel Mining). FPM has the same simple communication scheme as CD, i.e., only the support counts are exchanged. In order to do pruning by the two pruning techniques, FPM needs more information than CD. Distributed pruning needs the gl-large marks at each processor of all large itemsets, while global pruning needs the local support counts. Note that these information doesn't cause any extra computation, as it is available at the end of the previous iteration.

The first iteration of FPM is the same as CD. Each processor scans its local partition to find the local support counts of all size-1 itemsets and uses one round of count exchange to compute the global support counts. At the end, in addition to L_1 , each processor i also has the gl-large itemsets $GL_{1(i)}$ and their local support counts.

For the k -th iteration of FPM, $k > 1$, the program fragment at processor i , is described in Fig. 2.

Both CD and FPM were implemented with the collective communication operations of MPI on the IBM SP2. In order to compare the effectiveness of distributed and global pruning , we have also implemented a variant FNG (FPM with No Global pruning) of FPM. FNG does not perform global pruning, i.e., step 2 in Fig. 2 is removed for FNG.

- 1) compute candidate sets $C_k = \bigcup_{i=1}^n \text{apriori_gen}(GL_{k-1(i)})$ (distributed pruning);
- 2) prune candidates in C_k by global pruning;
- 3) scan the local partition to find the local support counts $X_{\cdot sup(i)}$ for all remaining candidates $X \in C_k$;
- 4) exchange $\{X_{\cdot sup(i)} \mid X \in C_k\}$ with all other processors to get the global support counts $X_{\cdot sup}$, for all $X \in C_k$;
- 5) compute $GL_{k(i)} = \{X \in C_k \mid X_{\cdot sup} \geq \text{minsup} \times |D|, X_{\cdot sup(i)} \geq \text{minsup} \times |D_i|\}$.
- 6) return $L_k = \bigcup_{i=1}^n GL_{k(i)}$.

Fig. 2. The FPM Algorithm

4 Data Skewness and Pruning Effect

Both distributed and global pruning make use of the distribution of itemsets among database partitions to do pruning. The effectiveness of the two pruning techniques depends on the itemset distribution, which can be captured as *data skewness*. We first give a formal definition for data skewness, and then evaluate the dependence of the two pruning techniques on data skewness.

4.1 Data Skewness Metric

We have developed a data skewness metric based on the well established notion of entropy [5]. For a random variable, the entropy is a measure of the non-uniformity of its probability distribution. Suppose a database is partitioned into n partitions D_1, D_2, \dots, D_n . For an itemset X , let $X_{\cdot sup}$ and $X_{\cdot sup(i)}$ be the global support counts and the local support counts of X at processor i , respectively. The value $p_X(i) = \frac{X_{\cdot sup(i)}}{X_{\cdot sup}}$ can be regarded as the probability of occurrence of an itemset X in partition D_i , ($1 \leq i \leq n$), as $\sum_{i=1}^n p_X(i) = 1$. The entropy $H(X) = -\sum_{i=1}^n (p_X(i) \times \log(p_X(i)))$ is a measure of the distribution of the local support counts of X among the partitions. We first define the skewness of the distribution of an itemset.

Definition 3. Given a database partition D_i ($1 \leq i \leq n$). The skewness $S(X)$ of an itemset is defined by

$$S(X) = \frac{H_{max} - H(X)}{H_{max}}, \text{ where } H(X) = -\sum_{i=1}^n (p_X(i) \times \log(p_X(i))) \text{ and } H_{max} = \log(n).$$

The skewness $S(X)$ has the following properties:

- $S(X) = 0$, when all $p_X(i)$, $1 \leq i \leq n$, are equal. So the skewness is at its lowest value when X is distributed evenly in all partitions.
- $S(X) = 1$, when a $p_X(i)$ equals to 1 and all the others are 0. So the skewness is at its highest value when X occurs only in one partition.
- $0 < S(X) < 1$, in all the other cases.

Then we define the skewness of a database partition as a weighted sum of the skewness of all itemsets.

Definition 4. Given a database partition D_i , $(1 \leq i \leq n)$, the data skewness $TS(D)$ of the partition is defined by

$TS(D) = \sum_{X \in IS} S(X) \times w(X)$, where IS is the set of all itemsets, $w(X) = \frac{X_{sup}}{\sum_{Y \in IS} Y_{sup}}$, and $S(X)$ is the skewness of X .

$TS(D)$ has some properties similar to $S(X)$.

- $TS(D) = 0$, when the skewness of all itemsets are at its minimal value.
- $TS(D) = 1$, when the skewness of all itemsets are at its maximal value.
- $0 < TS(D) < 1$, in all the other cases.

We can compute the data skewness of a partition following the metric defined in Definition 4. However, the number of itemsets may be very large in general. One refinement is to compute the skewness over the set of globally large itemsets only, and reduce $p_X(i) = 0$ if X is not gl-large in D_i . This refinement is a practical and reasonable approximation. (1) In generating candidates, only globally large itemsets will be joined together to form new candidates. Hence, only their skewness would impact the effectiveness of pruning. (2) The number of large itemsets is usually much smaller than all itemsets.

Table 1 shows an example for computing data skewness in two cases with high and low data skewness respectively. Suppose the global and the local support thresholds are 15 and 5 respectively.

		Items	A	B	C	D	E	F
Case 1 high data skewness	local count at processor 1	13	33	1	2	2	1	
	local count at processor 2	1	3	12	34	1	4	
	local count at processor 3	2	1	2	1	12	33	
	$S(X)$	1	1	1	1	1	1	
	$TS(D)$					1		
Case 2 low data skewness	local count at processor 1	6	12	4	13	5	12	
	local count at processor 2	6	12	5	12	4	13	
	local count at processor 3	4	13	6	12	6	13	
	$S(X)$	0.369	0.001	0.373	0.001	0.373	0.001	
	$TS(D)$					0.087		

Table 1. Data skewness computation based on gl-large itemsets

4.2 Pruning Effect and Data Skewness

Table 1 shows two cases with high and low data skewness respectively. In case 1 with high data skewness, items A and B , C and D , E and F are distributed mostly at processors 1, 2 and 3 respectively. In this case, apriori_gen will generate $\binom{6}{2} = 15$ candidates at the second iteration, while distributed pruning will generate only three candidates AB , CD and EF , as only these itemsets have a processor at which all their subsets are gl-large.

The following theorem states the minimum number of candidates that distributed pruning could generate at the second iteration if the database partition has high data skewness.

Theorem 5. *Let L_1 be the set of size-1 large itemsets, $C_{2(a)}$ and $C_{2(d)}$ are size-2 candidates generated by apriori_gen and distributed pruning, respectively. Suppose that each size-1 large itemset is gl-large at one and only one processor, and the number of size-1 gl-large itemsets at each processor is $\frac{|L_1|}{n}$, where n is the number of processors. Then $\frac{|C_{2(d)}|}{|C_{2(a)}|} = \frac{\frac{|L_1|}{n}-1}{|L_1|-1} \approx \frac{1}{n}$.*

Proof: Since apriori_gen will generate all combinations in L_1 as size-2 candidates, $|C_{2(a)}| = \binom{|L_1|}{2} = \frac{|L_1| \times (|L_1|-1)}{2}$. As for distributed pruning, each processor will generate candidates independently from the gl-large size-1 itemsets at the processor. The total number of candidates it will generate is $|C_{2(d)}| = \left(\frac{|L_1|}{n}\right) \times n = \frac{|L_1|}{2} \times \left(\frac{|L_1|}{n} - 1\right)$. Therefore, $\frac{|C_{2(d)}|}{|C_{2(a)}|} = \frac{\frac{|L_1|}{n}-1}{|L_1|-1} \approx \frac{1}{n}$. \square

Theorem 5 shows that distributed pruning can dramatically prune away almost $\frac{n-1}{n}$ size-2 candidates generated by apriori_gen in the high data skewness case.

Case 2 of Table 1 is an example of low data skewness. Items A , B , C , D , E and F are equally distributed to the 3 processors. In this case, both apriori_gen and distributed pruning will generate the same 15 candidates.

Therefore, the effectiveness of distributed pruning depends on data skewness: it favors high data skewness and degrades to the level of apriori_gen in low skewness cases.

We have also examined the effectiveness of global pruning. As shown in Theorem 2, it subsumes distributed pruning. Table 1 also shows the effectiveness of global pruning. In the high data skewness case, global pruning can further prune away the two candidates CD and EF , besides those that are pruned away by distributed pruning. As for the low data skewness case, global pruning can also prune away the candidates AC , AE and CE .

From the example, it can be observed that both distributed and global pruning are sensitive to data skewness. They are very effective when data skewness is high. Also global pruning is more effective than distributed pruning and can perform significant pruning even in the mild data skewness case.

5 Performance Studies

The performance studies were carried out on an IBM SP2 with 32 nodes. Each node consists of a POWER2 processor with a CPU clock rate of 66.7MHz and 64 MB of main memory, and is running the AIX operating system. The processors communicate with each other through a high performance switch with an aggregated peak bandwidth of 40 MBps and a latency about 40 microseconds. The data was preloaded to the local disk of each processor in order to reduce the I/O time. We use synthetic databases to do the experiments and the database on each node is about 100MB in size.

5.1 Synthetic Databases Generation

The synthetic databases used in our experiments are generated using similar techniques introduced in [2]. We have enhanced it to generate data partitions with a parameter to control data skewness.

We generate the synthetic database partition by partition. All partitions are generated from a pool of potentially large itemsets as in [2]. We first generate the relative weights of these large itemsets. These weights are then broken down into weights in each partitions. In other words, every itemset in the pool has n weights associated with it instead of only one weight. Each one of these weights corresponds to the probability of occurrence of the itemset in a partition. The weight of each itemset in the size-L pool is picked from an exponential distribution with unit mean. Then we pick a skewness level s for the itemset from a normal distribution with mean S and variance 0.1. After that we generate n probability values from an exponential distribution with variance equal to s , and normalize them so that their sum equals to 1. We check the skewness of these n probability values with the metric in Definition 3. For each itemset, we repeat this random process until a set of n values, whose the skewness falls into the permitted range of $s \pm 0.02$, is generated. Then, these n probability values are randomly assigned to the n partitions. Eventually, the weight of the itemset is broken down into n weights by multiplying the weight of the itemset with the n probability values. The above procedure is iterated to generate the weights of all itemsets and their breakdowns. The second step is to generate the itemsets in the pool. We first divide the N items into n disjoint equal ranges. In order to control the skewness of itemsets, we regard the i -th ($1 \leq i \leq n$) probability values generated for an itemset in the previous step as the probability of choosing items from the i -th range to put into the itemset. All items of the first itemset are chosen randomly from some ranges, which are determined by tossing an n -sided weighted coin, where the weight of side i is the i -th probability of the n probability values of the itemset. Items are picked until the number is equal to the size of the itemset. Some items of the subsequent itemsets are copied from the previous itemset according to the correlation level as in [2], while the remaining items are picked in the same way as in the first itemset. The procedure to generate the transactions in the partitions from the itemsets is similar to that in [2].

5.2 Pruning Effect and Relative Performance

Fifteen databases have been generated. The name of the database is in the form Dx.Ty.Iz.Sr, where x is the number of transactions in each partitions, y is the average size of the transactions, z is the average size of the itemsets, and r is data skewness percentage. The number of partitions is 16, i.e., $n = 16$. We ran the three algorithms(FPM, FNG and CD) on 16 processors and measured the actual data skewness of the database partitions when running the programs. The measured data skewness of all databases are very closed to the control values.

The three figures on the left in Fig. 3 provide us with the pruning effect of distributed and global pruning on the fifteen databases. It shows the ratios of the number of candidates of FPM and FNG to that of CD. When the data skewness is high, ($S = 0.9$), distributed pruning has a 77.2% to 79.4% of reduction in candidates compared to *apriori_gen*, and global pruning has a 90.9% to 93.3% reduction. When data skewness is mild, ($S = 0.1$), distributed pruning only has a 3.6% to 6.8% reduction, but global pruning still has a 30.7% to 31.7% reduction. In all other cases with intermediate data skewness, the candidate pruning effect of the two pruning techniques are between the two extreme cases. These results confirm our observation on the effect of data skewness on the two pruning techniques, i.e., both distributed and global pruning favor partitions with high data skewness, and global pruning is more powerful than distributed pruning and remains to be effective even for partitions with mild data skewness.

The three figures on the right in Fig. 3 show the response times of the three algorithms on the fifteen databases. The results are very encouraging. In the more skewed cases, i.e., $S = 0.9$, FPM is 110% to 221% faster than CD, whereas FNG is 79% to 99% faster. In the less skewed cases, i.e., $S = 0.1$, FPM is 27% to 50% faster than CD, whereas FNG 3% to 5%. In intermediate data skewness cases, the performance improvement of FPM and FNG over CD is also mild. These results clearly demonstrate that the candidate reduction by distributed and global pruning leads to super performance of FPM over CD.

5.3 Parallelism : Speedup, Scaleup and Sizeup

In order to show the advantage of the parallelism, we investigated the performance of the three parallel algorithms in speedup, scaleup and sizeup experiments. We used the database D3278K.T5.I2.S70, which has high data skewness, as the base database to perform the experiments.

In the speedup experiment, we investigated the performance speedup on a fixed size database with increasing number of processors and partitions. These databases have 16 partitions with a total size of 1.6GB. We combined the partitions to form databases with 8, 4, 2, and zero partitions. With this configuration, we perform speedup comparison on 1, 2, 4, 8, and 16 processors.

The first two figures in Fig. 4 show the execution times of the three algorithms and their speedups. In this case, FNG almost achieves the ideal linear speedup. As for FPM, its speedup is superlinear. The reason behind this superlinear speedup is the increasing pruning power of the two pruning techniques with the number of partitions increasing.

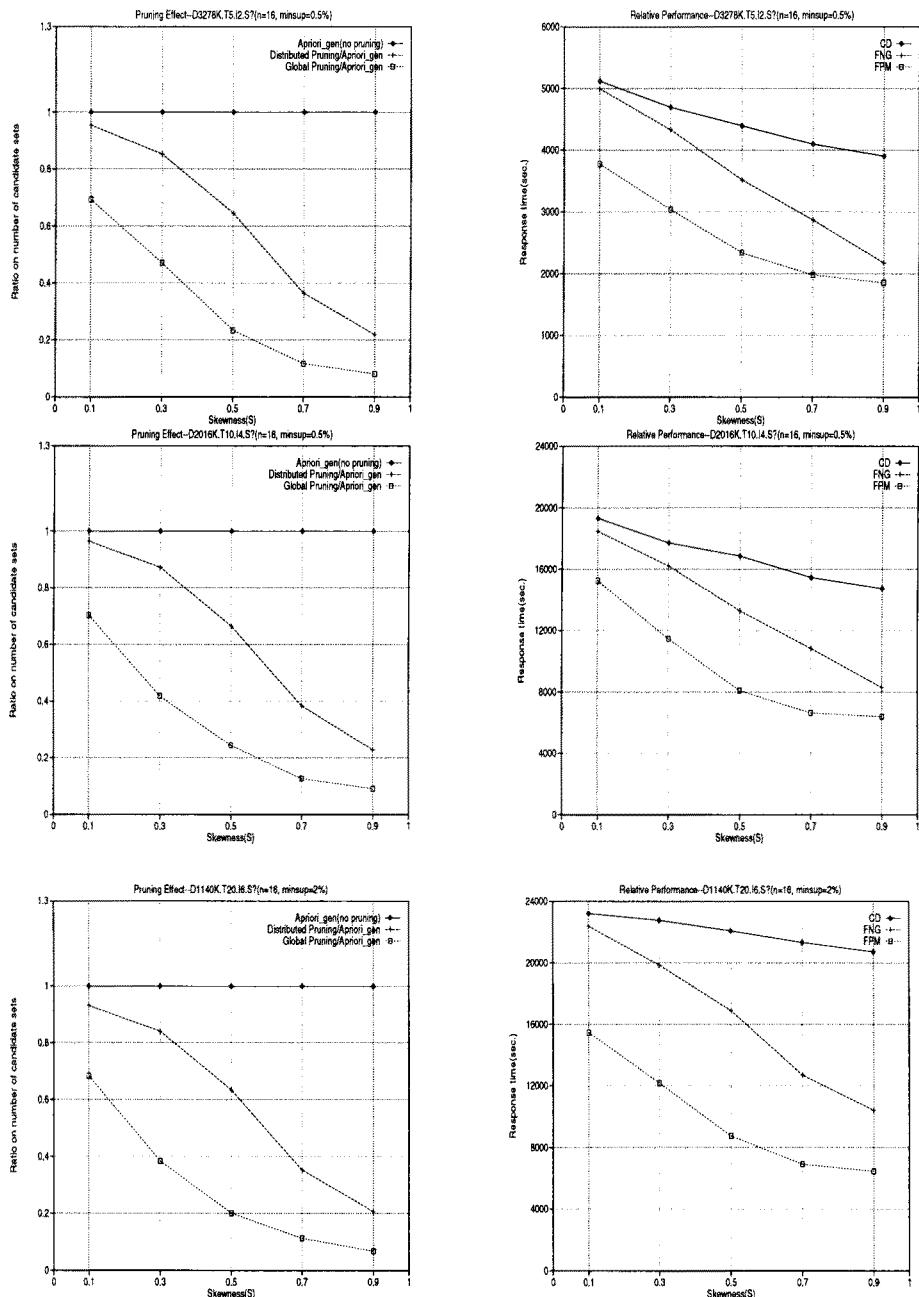


Fig. 3. Pruning Effect and Relative Performance

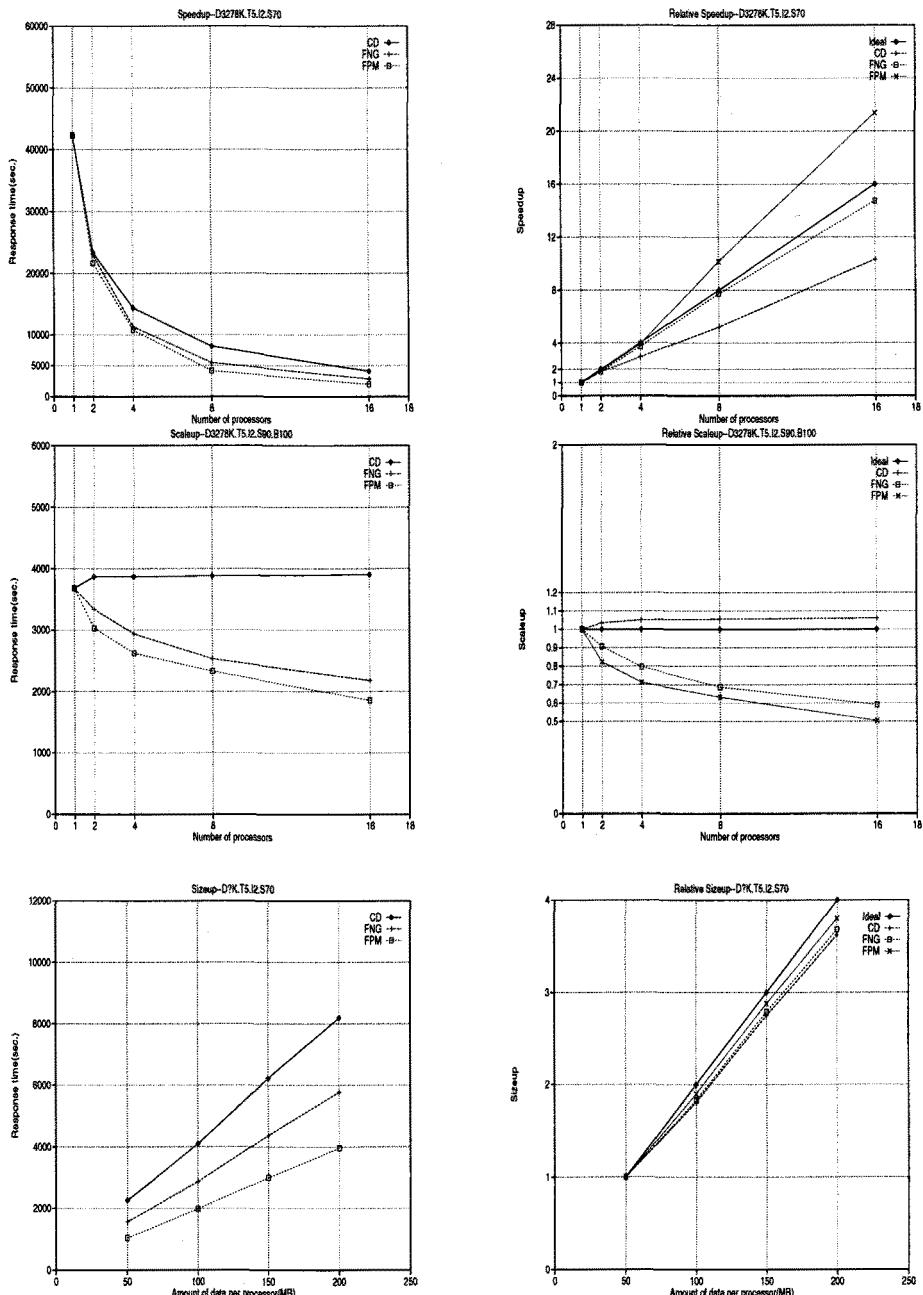


Fig. 4. Speedup, Scaleup and Sizeup

In the scaleup experiment, we generated databases of different sizes. The sizes of our databases range from 100MB to 1.6MB. The number of processors involved were also increased from 1 to 16 correspondingly. Each database were partitioned on a number of processors such that every partition is equal to 100MB. Therefore, both database size and number of processors are scaled up proportionally. In the generation of the databases and their partitions, we used the parameters and control values in D3278K.T5.I2.S70. Note that even though each database has the same control value on data skewness, their actual data skewness are larger if they have more partitions. The third and the fourth figures in Fig. 4 show the scaleup results. As expected, CD is able to keep its response time almost constant when both data size and number of processors increase linearly, while the response times of FPM and FNG decrease due to higher candidate set size reduction when the number of partitions increases. This also shows that FPM has very small amount of overhead.

For the sizeup experiments, we used the parameters and control value in D3278K.T5.I2.S70 to generate different databases with the same fixed number of partitions but different total size. The number of processors involved is fixed to $n = 16$. The size of each partition ranges from 50MB to 200MB. The last two figures in Fig. 4 show that all three algorithms have a sublinear performance. This is due to the fact that the computation time and disk I/O time are proportional to the size of the databases. Again FPM is much more close to the ideal linear size up than CD.

6 Conclusion

In this paper, we have proposed a parallel algorithm FPM for mining association rules. Our performance studies on an IBM SP2 parallel machine have shown that FPM outperforms CD consistently. It also has nice parallelism of scaleup, speedup and sizeup.

The edge of FPM over CD is the incorporation of distributed and global pruning techniques. The effectiveness of these two techniques is related to data skewness of the partitions. We have developed a metric to measure the data skewness of a partition based on the notion of entropy. Our analysis and experiment results show that distributed pruning favors data partitions with high data skewness. More importantly, global pruning is more powerful than distributed pruning and very effective even for partitions with mild data skewness.

FPM adopts the count distribution approach to compute large itemsets. It has a simple and light-weight communication scheme. The pruning technique provides a lot of savings in searching and communication because of the much smaller candidate sets compared to CD. In the future work, we are interested to adopt the pruning techniques to develop parallel mining algorithms on a shared-memory parallel system. Another important problem is to develop techniques to partition a database such that the result has high data skewness.

References

1. R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proc. of 1993 ACM-SIGMOD Int. Conf. On Management of Data*, Washington, D.C., 1993.
2. R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proc. of the 20th VLDB Conference*, Santiago, Chile, 1994.
3. R. Agrawal and J.C. Shafer. Parallel mining of association rules: Design, implementation and experience. Special Issue in Data Mining, *IEEE Trans. on Knowledge and Data Engineering*, IEEE Computer Society, V8, N6, December 1996, pp. 962-969.
4. S. Brin, R. Motwani, J. Ullman, S. Tsur. Dynamic itemset counting and implication rules for market basket data. In *Proc. of 1997 ACM-SIGMOD Int. Conf. On Management of Data*, pages 255-264, Tucson, Arizona, 1997.
5. T. M. Cover, T. A. Thomas. Elements of information theory. John Wiley & Sons, Inc., 1991.
6. D. W. Cheung, J. Han, V. Ng and C. Y. Wong. Maintenance of discovered association rules in large databases: An incremental updating technique. In *Proc. of the 12th Int. Conf. on Data Eng.*, New Orleans, Louisiana, 1996.
7. D. W. Cheung, J. Han, V. T. Ng, A. W. Fu, Y. Fu. A fast distributed algorithm for mining association rules. In *Proc. of 4th Int. Conf. on Parallel and Distributed Information Systems*, pages 31-43, Miami Beach, Florida, December, 1996.
8. E. Han, G. Karypis and V. Kumar. Scalable parallel data mining for association rules. In *Proc. of 1997 ACM-SIGMOD Int. Conf. On Management of Data*, pages 277-288, Tucson, Arizona, 1997.
9. J Han and Y Fu. Discovery of multiple-level association rules from large databases. In *Proc. of the 21th VLDB Conference*, Zurich, Switzerland, 1995.
10. Message Passing Interface Forum. MPI: A Message-Passing Interface Standard, May 1994.
11. J. S. Park, M. S. Chen, and P. S. Yu, An effective hash-based algorithm for mining association rules. In *Proc. of 1995 ACM-SIGMOD Int. Conf. on Management of Data*, pages 175-186, San Jose, CA, May 1995.
12. J. S. Park, M. S. Chen, and P. S. Yu, Efficient parallel mining for association rules. In *Proc. of the 4th Int. Conf. on Information and Knowledge Management*, Baltimore, Maryland, 1995.
13. A. Savasere, E. Omiecinski, and S. Navathe. An efficient algorithm for mining association rules in large databases. In *Proc. of the 21th VLDB Conference*, Zurich, Switzerland, 1995.
14. R. Srikant and R. Agrawal. Mining quantitative association rules in large relational tables. In *Proc. 1996 ACM-SIGMOD Int. Conf. on Management of Data*, Montreal, Canada, 1996.
15. T. Shintani, M. Kitsuregawa. Hash based parallel algorithms for mining association rules. In *Proc. of 4th Int. Conf. on Parallel and Distributed Information Systems*, Miami Beach, Florida, 1996.
16. M. J. Zaki, M. Oghara, S. Parthasarathy, and W. Li, Parallel data mining for association rules on shared-memory multi-processors. *Supercomputing'96*, Pittsburg, PA, Nov 17-22, 1996.

Trend Directed Learning: A Case Study

Honghua Dai*

Department of Software Development

Monash university

Caulfield, Vic 3145, AUSTRALIA

Tel: (02) 6773.3182 Fax: (02) 6773.3312

hdai@turing.une.edu.au

Honghua.Dai@fcit.monash.edu.au

Abstract

Misleading caused by low quality data is a well known problem in knowledge discovery in databases. Several techniques have been introduced to deal with the problem which include inexact learning strategies, such as rough set based approaches and probabilistic approaches. This paper presents an approach for detecting trend using contribution functions. A trend directed method for the discovery of knowledge structure from low quality data bases is described. The experimental results show that trend directed methods are superior to other learning strategies, particularly when the learning is performed on low quality data bases.

Keywords: Knowledge discovery, noisy data, trend detection, data mining, machine learning, inexact learning, knowledge acquisition.

1 Introduction

A general concern in knowledge discovery in databases is the reliability of the derived rules. This problem is very essential to all learning algorithms. In a normal case, most learning algorithms can do very well on high quality data sets. But only a few learning algorithms can achieve a high predictive accuracy rate on low quality data sets. As a consequence, the discovered rules may make no sense at all or the data be totally misinterpreted. The predictive accuracy rate of the rules is normally very low. Even though the derived rules may still fit the training data set very well. But it is not the real nature of the data.

The reliability[5] of derived rules could be affected by many factors including the quality of data and the robustness of a learning algorithm.

Some research work has been done in this area. A number of approaches have been proposed to deal with this problem for avoiding misleading caused by low quality data for the achievement of a better predictive accuracy. In 1994, Dai and Ciesielski (Dai and Ciesielski, 1994) [2, 3, 4, 6] pointed out the low quality data (LQD) problem and introduced a rough set theory[7, 8, 9] based inexact field learning approach which has been proven that a

*This work was done at Monash university. At the time of preparing this final copy, the author has been employed by the School of Math and Computer Science, the University of New England at Armidale, NSW 2351, Australia.

higher accuracy rate can be achieved by this approach. The method is robust even when it is provided with data sets with noisy, missing values and redundant features.

In the section 2 of this paper, we explain our motivation of this research. In section 3, we address the possible problems in knowledge discovery in low quality databases. In section 4, we give a description of *Trend* and the *Trend Detection*, look at the fundamental issues and the conditions of trend detection. In section 5 we present the trend directed learning and consider the advantages of applying trend as a guide in further knowledge discovery process. The last section gives a conclusion of the work.

2 The Motivation

In computer science, there are many examples where difficult problems be solved with the aid of some kind additional information, such as heuristic search in Artificial Intelligence which is an efficient way to reduce search space and to enhance the search speed. In general case the complexity of a heuristic search method is much lower than an ordinal search method. The problem is, how to find out a good heuristic function. Another good example of using additional information in solving difficult problems is knowledge based systems techniques. Again, the central problem is, how to discover useful knowledge. This is the central task of machine learning, knowledge discovery and data mining. An interesting question is that could we consider a heuristic learning or knowledge based learning or learning aided learning? Basically, it is possible. Actually people have been trying to do so for some period. What we consider here is to learn something which is similar to a heuristic function before the learning of a target rule. The idea for doing so is to avoid possible misleading which could be caused by a low quality data set. Such function (or in other form) is then applied as a guide in the learning of the target rule. Such kind function (or in other form) is called a *trend* which is a rough rule and is potentially the true reflection of a target rule.

Before we give a definition of a *Trend* and discuss the strategies to find a *trend* and presents the ways to use a *trend* in learning, we start with a case study which shows the real possibilities of a learning process.

3 Incompleteness, Misleading and Approximation of Learning

Given a data set, in some cases a learning algorithm may not be able to find a useful rule which is a true interpretation of the data. Even a rule is found. when you apply the rule to a real problem to make prediction or classification, it is very common that the rule found by one algorithm makes better prediction than those discovered by the others'. We call the first problem the **case dependent problem** of learning and the second problem the **algorithm dependent problem**.

To find the best rule, let us look at an example first. Given a data set as shown in Table 1, we use 1 to represent the state *Solid* and -1 the *Liquid*. In a real knowledge discovery process, a learning algorithm could come across with various problems. Among these problems, incomplete knowledge and misleading rules are the most significant ones. These two problems basically are case dependent problems.

Temperature (T)	State
-8°C	1 (Solid)
-4°C	1 (Solid)
-2°C	1 (Solid)
-1°C	-1 (Liquid)
2°C	1 (Solid)
4°C	-1 (Liquid)
8°C	-1 (Liquid)
9°C	-1 (Liquid)

Table 1: An example data set

Incomplete knowledge

Incomplete knowledge problem refers to the problem that the knowledge discovered by a learning system is unable to make decision in some circumstance if it is applied to solve a certain task. Such knowledge is called incomplete knowledge.

In a learning process, a learning algorithm may turn up with two rules:

Rule 1.

$$\text{If } (T \leq -2^{\circ}\text{C}) \text{ Then } 1(\text{Solid}) \quad (1)$$

Rule 2.

$$\text{If } (T \geq 4^{\circ}\text{C}) \text{ Then } -1(\text{Liquid}) \quad (2)$$

In this case, if the temperature $T \in (-2^{\circ}\text{C}, 4^{\circ}\text{C})$, then the rules will not be able to make any conclusion. If this happens, we say the learning algorithm caused an *incompleteness problem* in its discovery process. That is to say, the discovered rules do not apply to part of the domains. This is a very common problem in knowledge discovery in databases.

Misleading rules

The second possible problem in knowledge discovery in databases is misleading rules. A discovered rule is called a misleading rule if the rule contains misleading information.

Suppose a learning algorithm turn up with a rule as following,

Rule 3.

$$\begin{aligned} \text{If } (T \leq 2^{\circ}\text{C}) \text{ Then } 1(\text{Solid}) \\ \text{Otherwise } -1(\text{Liquid}) \end{aligned} \quad (3)$$

or

Rule 4.

$$\begin{aligned} \text{If } (T \geq -1^{\circ}\text{C}) \text{ Then } -1(\text{Liquid}) \\ \text{Otherwise } 1(\text{Solid}) \end{aligned} \quad (4)$$

then we say it is a misleading rule. Misleading rule is derived by a learning algorithm due to the misleading of low quality data and for the lacking of processing facility for the preventing of misleading. There are several strategies which have been proposed to deal with misleading problem. One successful experiment is to induce inexact rules[2] from low quality data rather than exact rules.

Inexact rules (Approximation)

Obviously, both the incompleteness problem and the misleading problem are caused by the quality of the data and perhaps the ability of the algorithm.

One strategy which was proposed by (Dai and Ciesielski, 1994)[2] is to come up with an inexact rule rather than exact rules. By this method, a rule as follows can be derived,

$$\text{If } T = \begin{cases} \leq -2^{\circ}C & \text{Then } 1(\text{Solid}) \\ \geq 4^{\circ}C & \text{Then } -1(\text{Liquid}) \\ \in (-2^{\circ}C, 4^{\circ}C) & \text{Then } \frac{1-T}{3} \end{cases} \quad (5)$$

This rule (5) has extended the rule set $S = \{Rule1, Rule2\}$ by adding the rule as below,

Rule 5:

$$\text{If } (T \in (-2^{\circ}C, 4^{\circ}C)) \text{ Then State} = \frac{1-T}{3} \quad (6)$$

The Rule 5 is called an inexact rule which gives a rough classification. Such inexact rules are very useful when the quality of the data is low. Look at the data table 1, the error items were,

Temperature (T)	State
$-1^{\circ}C$	-1 (Liquid)
$2^{\circ}C$	1 (Solid)

Table 2: The data items with errors

By using the derived inexact rule, we can get better data items,

Temperature (T)	State
$-1^{\circ}C$	$\frac{2}{3}$ (83% Solid)
$2^{\circ}C$	$-\frac{1}{3}$ (67% Liquid)

Table 3: The Improved data items

If we replace the old data items (with error) with these improved data items, we can get an improved data set,

Temperature (T)	State
-8°C	1 (Solid)
-4°C	1 (Solid)
-2°C	1 (Solid)
-1°C	$\frac{2}{3}$ (83% Solid)
2°C	$-\frac{1}{3}$ (67% Liquid)
4°C	-1 (Liquid)
8°C	-1 (Liquid)
9°C	-1 (Liquid)

Table 4: An Improved example data set

This suggests that the derived inexact rule can be applied for the improvement of the data quality. But this is not our main interests.

Our major interests is to use this derived inexact rule to direct further learning process in the hope of finding more accurate rules.

Figure 1 illustrates an example of a trend function which is derived from the data set as shown in Table 1 using contribution function. It is a continuous piecewise smooth function. The interval $[-2, 4]$ is called a non-deterministic field or a rough field.

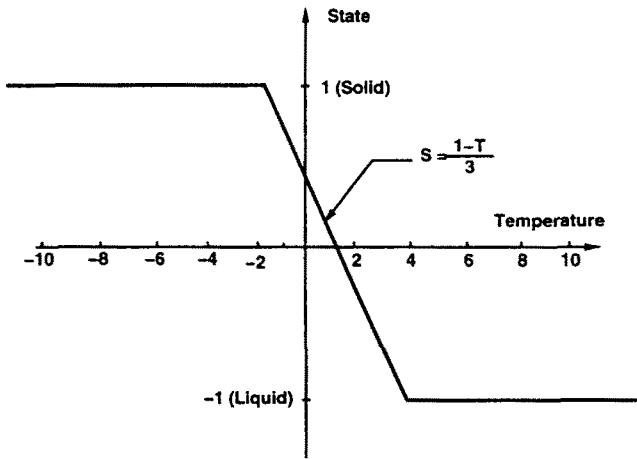


Figure 1: An Example of a Trend Function

4 Trend and Trend Detection

It is very important to notice that if a rule is implied in a data set, then we should be able to find the rule by some means and second, a trend should also be found.

Definition 4.1 Trend.

A trend is a preliminary knowledge structure which is derived from the given data set, and is in a rough form of the target knowledge structure. It reflects the potential or the possibility of the target knowledge to be discovered.

According to the definition, a trend must be: (1) a preliminary rule; (2) able to describe the potential or the possibility of a target rule to be discovered; (3) derived from a given data set, i.e., the training data set; (4) can be used to direct further learning process towards a better rule. Trends could be in various forms. It can be an inexact rule [3], a table, a frame, a decision tree, a probability description, a possibility description, a function, a specification, a descriptive complexity description including MML/MDL, computational complexity description or weight description. An example of a trend is a contribution function[2, 3] which provides a measurement of the degree in giving correct classification by an ideal target rule.

The detection of a trend is based on the following assumptions: (1) there exists at least one rule which is a reasonable good reflection of the given data set; (2) the rule can be discovered by a learning algorithm; (3) several competitive rules may exist. In such case, a trend must exists and can be found.

The process of trend detection itself is a learning process. Unlike normal learning process, it is not necessary to be precise. However it must be qualified to reflect the true potential or the possibility of being a rule or a model and can be further applied to aid the learning of a more accurate knowledge structure.

Let's look at an example. In this example, the trend is represented as a contribution function. A contribution function is a special case of a relevance function. In [3], Dai and Ciesielski gave a definition of relevance function as follows,

Definition 4.2 Relevance Function

Let $X \subseteq U$, U be a set called universe, and let R be a binary relation over U . The pair $A = (U, R)$ is called an approximation space, the relevance function is a rough set membership function which is defined as follows[7, 9],

$$\mu_X(x) = \begin{cases} 1 & \text{iff } x \in \underline{A}(X) \\ \frac{1}{2} & x \in Bn_A(X) \\ 0 & \text{iff } x \in U - \bar{A}(X) \end{cases} \quad (7)$$

where $\underline{A}(X)$ is the lower approximation of X in A , $\bar{A}(X)$ is the upper approximation of X in A and the set $Bn_A(X) = \bar{A}(X) - \underline{A}(X)$ is the boundary of X in A .

In formula (7), “0” represents “not relevant”, “1” represents “100% relevant”.

In our previous work we extend the definition of relevance function and modify formula (7) as follows:

$$\mu_X(x) = \begin{cases} 1 & \text{iff } x \in \underline{A}(X) \\ \frac{x-b}{a-b} & x \in Bn_A(X) \\ 0 & \text{iff } x \in U - \bar{A}(X) \end{cases} \quad (8)$$

where $a = \|\underline{A}(X)\|_{(U - \bar{A}(X))}$ denotes that a is a point in $\underline{A}(X)$ i.e., $a \in \underline{A}(X)$ and under certain measurement $\|\cdot\|$, for all $x \in \underline{A}(X)$ and $y \in U - \bar{A}(X)$, the a satisfy $\|a - y\| \leq \|x - y\|$. The same for $b = \|U - \bar{A}(X)\|_{\underline{A}(X)}$.

In this paper the term relevance function is specifically used for measuring the degree of the relevance between a case I_i ($1 \leq i \leq n$) and a class c_k ($1 \leq k \leq s$).

A contribution function is a function to be applied for measuring the strength of the relationship of an attribute x_j ($1 \leq j \leq n$) to the output variable γ . In other words, a contribution function gives the measurement of the degree of the support of an attribute to a particular class. That is, the contribution function is a function of a attribute. A contribution function is defined as follows:

Definition 4.3 Contribution Function

Let $X = \{x_1, x_2, \dots, x_n\}$ be a set of attributes, γ be the output variable, $A_r \subset X$ be the set of attributes which support γ , and $\bar{A}_r \subset X$ be the set of attributes which deny γ , the contribution function is defined as follows

$$\nu_{A_r}(x) = \begin{cases} 1 & \text{iff } x \in A(X_r) \\ \frac{x-b}{a-b} & x \in \bar{A}(X_r) - A(X_r) \\ 0 & \text{iff } x \in X - \bar{A}(X_r) \end{cases} \quad (9)$$

where $\bar{A}(A_r)$ and $A(A_r)$ are the upper approximation and the lower approximation of the subset A_r .

The function (9) is a trend, because first, $\nu_{A_r}(x)$ is an inexact rule in rough form. Secondly, it is a good approximation of the target rule to be discovered. And thirdly, it potentially reflects the possible target rule.

5 Trend Directed Learning

In a simple word, trend directed learning is the learning process which is directed by the pre-found trend. In other word, we use the profound trend as the heuristics to guide the induction of a target rule.

As we explained earlier, a trend could be in one representation and the other trends which are applied in other trend directed learning process could be in another representation. Naturally, the form of a trend will affect the process of a trend directed learning algorithm. In another word, trend directed learning is trend dependent which is very similar to a heuristic search algorithm in artificial intelligence.

Previous Trend Directed Learning

Some of the existing learning algorithms are trend directed learning. These algorithms use the trend which is designed by human beings rather than derived by the learning system itself. Some examples include MML criterion[13, 15, 14, 16],

$$L = -\log_2 P(H) - \log_2 P(D|H) = L(H) + L(D|H) \quad (10)$$

where $L(H) = -\log_2 P(H)$ is the cost (in number of bits) of encoding the theory H , and $L(D|H) = -\log_2 P(D|H)$ is the cost of encoding the sample data D given the theory H is true.

The function (10) is a trend according to our definition which can be applied to aid the induction of the theory.

The second example is ID3[10, 11, 12] which uses the formula,

$$I(p, n) = -\frac{p}{p+n} \log_2 \frac{p}{p+n} - \frac{n}{p+n} \log_2 \frac{n}{p+n} \quad (11)$$

to aid the construction of a decision tree. The formula (11) is a trend. The learning pattern of these algorithms is illustrated in Figure 2. The keypoint is that in this trend learning pattern the trend is well designed by human beings rather than by a learning process itself.

Automated Trend Directed Learning

Figure 3 illustrates an automated trend directed learning system. The strategies of a rough set theory based trend directed learning can be outlined as follows:

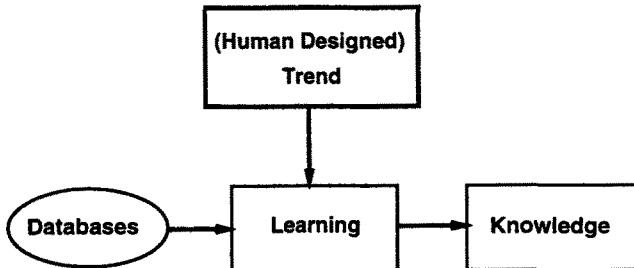


Figure 2: Traditional Trend Directed Learning Pattern

Step 1. Contribution function construction

For a given data set, a contribution function is constructed for each one of the attributes with respect to each class. An attribute could be excluded from further consideration if it is identified as irrelevant, or without or with very low contribution to a class.

Step 2. Trend Induction

Derive a trend using the strategy which is applied for the induction of an inexact rule as described in [1, 2, 3] or some other strategies which are identified as suitable. Such trend is a continuous piecewise smooth function with a rough field. This step can be viewed as a pre-learning of a target rule, or the first learning stage.

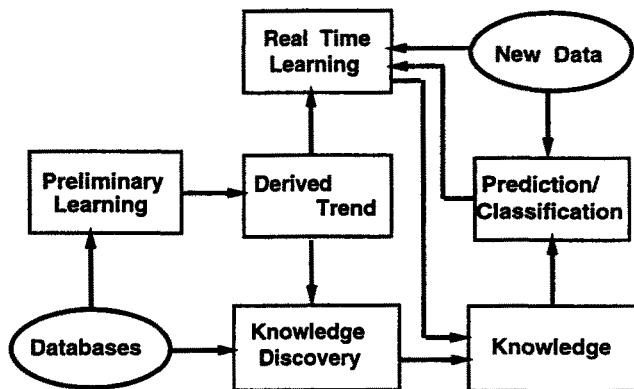


Figure 3: Automated Trend Directed Learning Pattern

Step 3. Rough Field Reduction

Each rough field is subdivided into three equal sized subfields. We examine the two sub-rough fields which related to the two classes. Merge one of the two sub-rough fields at a time, if this does not cause a reduction of the predictive accuracy, then the sub-rough field is

excluded from the original rough field, otherwise, further sub-division is taken. If no further rough field reduction can be done, a mid-point of the rough field is taken as the classification point. If this does not reduce the accuracy, an exact rule is found.

Step 4. Learning of Target Rules

An exact learning algorithm induces rules on the reduced rough field under the direction of the trend which provides the possibility of being a class. The output of this step is a target with an attached trend.

Step 5. Learning With the Aid of Trend

In the application of the learnt target rules, a real time learning which frequently checking the consistency of the trend, the performance of the target rule and update the target rule towards achieving more accurate rate.

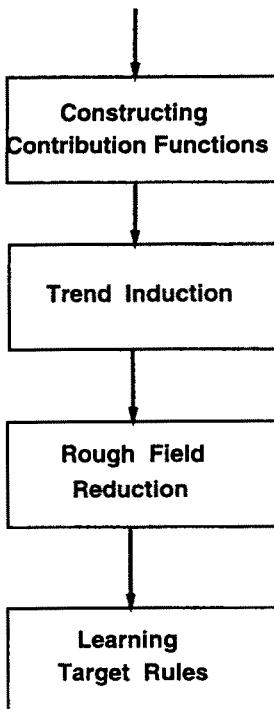


Figure 4: The Procedure of a Rough Set Based Trend Directed Learning

To explain our idea, let's look at the previous example given in the section 3. the third line of the formula (5), i.e., **Rule 5** which provided a good solution to the field where a vague case exists. The **Rule 5** gives an approximate fitting of the undefined domain $(-2^{\circ}c, 4^{\circ}c)$. It is the fact that the rule is partially controversial to the original data, that is in the interval $(-2^{\circ}c, 4^{\circ}c)$. the rule predicts that $state = \frac{1-T}{3}$, when $T \in (-2^{\circ}c, 4^{\circ}c)$ which gives $state = 1(Solid)$ at $T = -2^{\circ}c$ and gives $state = -1(Liquid)$ at $T = 4^{\circ}c$. This result is controversial with the data in which $state = 1(Solid)$ when $T = -1^{\circ}c$ and $state = -1(Liquid)$ when $T = 4^{\circ}c$. Such modification were given according to the rule 1 and 2.

In this one attribute case, the rough rule **Rule 5** is the trend and the field $(-2^0c, 4^0c)$ is the rough field. In step 3, we subdivide the rough field $(-2^0c, 4^0c)$ into three sub-rough fields $[-2^0c, 0^0c)$, $[0^0c, 2^0c)$ and $[2^0c, 4^0c)$.

Preliminary Experimental Results

Our experimental results show that (1) on low quality data sets, a rough rule can make a better prediction than an exact rule; and (2) a rule induced by a rough set theory based trend learning approach could achieve an even better predictive accuracy rate.

we can easily see that the rough rule **Rule 5** gives a better prediction when $T \in (-2^0c, 4^0c)$. It potentially approaches to the real rule.

The results we achieved did not implement the final step of the learning procedure. We expect that a better results could be achieved if the final step of the whole learning procedure is further implemented.

6 Conclusion and Further Work

The main idea of trend directed learning is, first to find a trend which then be applied to direct the further learning process towards a better rule. The preliminary experimental results show that this method seems better in making prediction, particularly when the quality of given data is low. The results reported in this paper is the initial result. We expect that some further results will turn up soon and the further details of the theoretical and practical issues of the methods need to be further studied.

Some further work we are going to investigate include: (1) the other trend directed learning strategies; (2) the comparison of different trend learning strategies; (3) integrated multi-trend learning, trend co-operation and trend planning.

References

- [1] Victor Ciesielski and Honghua Dai. FISHERMAN: a comprehensive discovery, learning and forecasting systems. In *Proceedings of 2nd Singapore International Conference on Intelligent System*, pages B297(1)–B297(6), Nov, 1994.
- [2] Honghua Dai. Learning of forecasting rules from large noisy meteorological data. *Ph.D. Dissertation, Department of Computer Science, RMIT*, 1994.
- [3] Honghua Dai and Victor Ciesielski. Learning of inexact rules by the FISH-NET algorithm from low quality data. In *Proceedings of 7th Australian Joint Conference on Artificial Intelligence*, pages 108–115. World Scientific, Nov, 1994.
- [4] Honghua Dai, Kevin Korb, and Chris Wallace. The discovery of causal models with small samples. In *Proceedings of the Fourth Australian and New Zealand International Conference On Intelligent Information Systems*, pages 18–20. IEEE, Inc., 18–20, November, 1996.
- [5] Honghua Dai, Kevin Korb, Chris Wallace, and Xindong Wu. A study of causal discovery with small samples and weak links. In *Proceedings of the 15th International Joint Conference On Artificial Intelligence*, pages 1304–1309. Morgan Kaufmann Publishers, Inc., 1997.

- [6] Honghua Dai, James Liu, and Vic Ciesielski. An analysis of criteria for the evaluation of learning performance. In *Proceedings of the Fourth Australian and New Zealand International Conference On Intelligent Information Systems*, pages 84–87. IEEE, Inc., 18-20, November, 1996.
- [7] Zdzislaw Pawlak. Rough sets. *International Journal of Information and Computer Sciences*, 11(5):145–172, 1982.
- [8] Zdzislaw Pawlak. Rough classification. *International Journal of Man-Machine Studies*, 20:469–181, 1984.
- [9] Zdzislaw Pawlak, S. K. M. Wong, and Wojciech Ziarro. Rough sets: Probabilistic versus deterministic approach. *International Journal of Man-Machine Studies*, 29:81–95, 1988.
- [10] Ross Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.
- [11] Ross Quinlan. *C4.5: Programms for Machine Learning*. Morgan Kaufmann Publishers, San Mateo, California, USA, 1993.
- [12] Ross Quinlan and Ronald L. Rivest. Inferring decision tree using the minimum description length principle. *Machine Learning*, August 10, 1987.
- [13] Chris Wallace and D.M. Boulton. An information measure for classification. *Computer Journal*, 11(2):185–194, 1968.
- [14] Chris Wallace and P. R. Freeman. Estimation and inference by compact coding. *Journal of the Royal Statistical Society, Series B*, 49(3):223–265, 1987.
- [15] Chris Wallace and Michael Georgeff. A general selection criterion for inductive inference. *ECAI 84, Advances in Artificial Intelligence*, pages 1–18, 1984.
- [16] Chris Wallace, Kevin Korb, and Honghua Dai. Causal discovery via MML. Technical report, Department of Computer Science, Monash University, 1996.

Interestingness of Discovered Association Rules in Terms of Neighborhood-Based Unexpectedness

Guozhu Dong Jinyan Li

The University of Melbourne, Department of Computer Science, Parkville, Vic 3052, Australia.
Email: {dong, jyli}@cs.mu.oz.au

Abstract. One of the central problems in knowledge discovery is the development of good measures of interestingness of discovered patterns. With such measures, a user needs to manually examine only the more interesting rules, instead of each of a large number of mined rules. Previous proposals of such measures include rule templates, minimal rule cover, actionability, and unexpectedness in the statistical sense or against user beliefs.

In this paper we will introduce neighborhood-based interestingness by considering unexpectedness in terms of neighborhood-based parameters. We first present some novel notions of distance between rules and of neighborhoods of rules. The neighborhood-based interestingness of a rule is then defined in terms of the pattern of the fluctuation of confidences or the density of mined rules in some of its neighborhoods. Such interestingness can also be defined for sets of rules (e.g. plateaus and ridges) when their neighborhoods have certain properties. We can rank the interesting rules by combining some neighborhood-based characteristics, the support and confidence of the rules, and users' feedback. We discuss how to implement the proposed ideas and compare our work with related ones. We also give a few expected tendencies of changes due to rule structures, which should be taken into account when considering unexpectedness. We concentrate on association rules and briefly discuss generalization to other types of rules.

1 Introduction

Data mining is concerned with the extraction of previously unknown and potentially useful high-level knowledge in the form of patterns from a huge mass of data. This area has received extensive attention from the research community and industry recently. When the amount of such high level knowledge is large, which is typically the case for association rules [1], the selection of interesting patterns becomes a serious problem for the human user; we will call this problem the post-mining rule analysis problem. Thus one of the central problems in the field of data mining is the development of good measures of interestingness of discovered patterns. With such measures, we can develop algorithms to help find the interesting rules from the mined rules. The results of mining can then be made more usable, without the need of going through all the mined rules manually. Previous proposals of interestingness measures include: rule templates [5, 4] for limiting attention to only those rules that match the templates, minimal rule covers [12] where rules implied by those presented to the user are eliminated, actionability of rules [8, 10] (some benefit can be obtained by doing something), and unexpectedness of rules [6, 11]. Unexpectedness has been interpreted either in the statistical sense, as

having higher chance than that under the independence assumption or as having higher chance than some threshold, or against user beliefs.

We believe that one should take the neighborhood of the rule into account when considering unexpectedness. Using mountains as an analogy, normally one would not say that all peaks of the Himalayas Range of height > 4000 meters are more interesting than the highest mountain in North America and Japan, although these peaks are higher than the highest mountain in North America and Japan. Indeed, the interestingness of a mountain depends on its height as well as on its position in its neighborhoods; indeed, Mount Fuji of Japan is famous because there are no comparable peaks in its neighborhood. In the terminology of association rules, the interestingness of a rule should depend on its confidence as well as on the degree of the confidence fluctuation in its neighborhoods and the density of mined rules there. The purpose of this paper is to introduce the neighborhood-based unexpectedness and to examine interestingness in terms of such unexpectedness.

We first present some novel notions of distance between rules and of neighborhoods of rules. The interestingness of a rule is then defined in terms of the pattern of the fluctuation of confidences or the density of mined rules in some of its neighborhoods. Such interestingness can also be defined for sets of rules (e.g. plateaus and ridges) when their neighborhoods have certain properties. We rank the interesting rules by combining some neighborhood-based characteristics, the support and confidence of these rules, and users' feedback. We discuss how to implement the proposed ideas and compare our work with related works. We also give a few expected tendencies of changes due to rule structures, which should be taken into account when considering unexpectedness. We will mainly concentrate on association rules and will briefly discuss generalization to other types of rules (by defining appropriate distance functions).

For the case of association rules, one might argue that it is possible to solve the post-mining rule analysis problem by increasing the thresholds – the number of mined rules will then decrease. Continuing with the world map analogy, one can easily see that as a result of such an approach only those global peaks will be shown to the user, thus missing the useful information conveyed by those local peaks over vast plains.

Our neighborhood-based interestingness is proposed as a complementary measure to those proposed previously in the literature, including those cited at the beginning of the introduction and the following. The issue of interestingness of general discovered knowledge was discussed in [7]. Interestingness can also be measured in terms of the statistical strength of a pattern such as confidence and support [2], and ancestor rules' confidences [9]. The problem of how to find patterns satisfying multiple criteria of interestingness has been investigated in [13]. A comprehensive survey about measurements of interestingness can be found in [11].

The rest of this paper is organized as follows. In Section 2, we present the preliminaries of association rules. In Section 3 we introduce the notion of distance among rules. In Section 4 we define neighborhoods of rules. In Section 5 we define interestingness of rules in terms of neighborhood-based unexpectedness, and similarly of sets of rules. In Section 6 we discuss some expected changes, and in Section 7 how to rank interesting rules. In Section 8 we cover some implementation issues. In Section 9 we compare our work with related ones. In Section 10 some concluding remarks are given.

2 Preliminaries on association rules

Let $I = \{i_1, i_2, \dots, i_n\}$ be a set of literals, called *items*. Let \mathcal{D} be a set of transactions, where each transaction¹ is a subset T of I . Given a set of items X from I , we say a transaction T *contains*, or *matches*, X if $X \subseteq T$; let $m(X)$ denote the set of *transactions* in \mathcal{D} which match X .

An association rule R is an implication of the form $X \rightarrow Y$, where $X \subseteq I$, $Y \subseteq I$, and $X \cap Y = \emptyset$. The *support* of R , denoted as $support(R)$, is $\frac{|m(XY)|}{|\mathcal{D}|}$: the percentage of transactions in \mathcal{D} which contain² XY . The *confidence* of R , denoted as $conf(R)$, is $\frac{|m(XY)|}{|m(X)|}$: the percentage of transactions containing X which also contain Y . (So confidence is defined only if $m(X) > 0$.)

The task of mining association rules is to find all association rules whose supports and confidences are larger than, respectively, some given minimum support threshold, *min_support*, and some minimum confidence threshold, *min_confidence*.

Given the item set I , there is a unique set of potential association rules. Given different thresholds for support and confidence, different sets of rules can be mined from a given set of transactions.

3 Distance definition

The central theme of this paper is to consider interestingness of rules in terms of neighborhood based unexpectedness. To this end, we need to have *some* distance functions between rules. Such distance can be defined in more than one way, including semantics-based ways and syntax-based ways; we will review a semantics-based distance and introduce a syntax-based one. The syntax-based definition has several parameters which can be adjusted to suit the need of particular applications.

3.1 A semantics-based distance

A semantics-based distance between association rules was given in [12]. It measures the difference between rules in terms of their sets of matching rows. More specifically, the *matching-set distance* between two rules $X_1 \rightarrow Y_1$ and $X_2 \rightarrow Y_2$ is defined as $|m(X_1 Y_1)| + |m(X_2 Y_2)| - 2 * |m(X_1 Y_1 X_2 Y_2)|$. Let $Dist_{mset}$ denote this function.

Using this definition, there can be different rules R_1 and R_2 , namely $A \rightarrow B$ and $B \rightarrow A$, such that $Dist_{mset}(R_1, R_2) = 0$. Thus $Dist_{mset}$ is not a metric distance over the set of all potential rules. This may lead to “more crowded” neighborhoods, and may have some effect on neighborhood-based unexpectedness.

3.2 A syntax-based distance

We now introduce a syntax-based distance, which is intended to measure the item-set difference between rules. This turns out to be a metric distance.

¹ A transaction is also referred to as a basket in the literature.

² Following the tradition of database literature and for the sake of clarity, we write XY for $X \cup Y$ where X and Y are sets of items. We use $|S|$ to denote the cardinality of a set S .

Our distance function is defined in such a way that one can give different scales of importance to differences for different parts of rules. Item-set differences are divided into three parts³: (i) the symmetric difference of all items in the two rules, (ii) the symmetric difference of the left-hand sides of the two rules, (iii) the symmetric difference of the right-hand sides.

Definition 1. Given three non negative real numbers $\delta_1, \delta_2, \delta_3$, define an *item-set distance* between two rules $R_1 : X_1 \rightarrow Y_1$ and $R_2 : X_2 \rightarrow Y_2$ as⁴

$$Dist_{iset}(R_1, R_2) = \delta_1 * |(X_1 Y_1) \ominus (X_2 Y_2)| + \delta_2 * |X_1 \ominus X_2| + \delta_3 * |Y_1 \ominus Y_2|.$$

Example 1. To illustrate this definition, consider the following rules:

$$\begin{array}{ll} R_1 : D \rightarrow BC & R_2 : AD \rightarrow BC \\ R_3 : BC \rightarrow D & R_4 : BC \rightarrow AD \end{array}$$

Then $Dist_{iset}(R_1, R_2) = \delta_1 + \delta_2$ and $Dist_{iset}(R_3, R_4) = \delta_1 + \delta_3$. Both of $\delta_1 + \delta_2$ and $\delta_1 + \delta_3$ are contributed by A , as items B, C , and D make no contribution to the two distances.

To illustrate the point that different “positional” differences make different contributions to the distance, consider the following rules:

$$R_5 : AB \rightarrow CD \quad R_6 : ADF \rightarrow CE$$

Then $Dist_{iset}(R_5, R_6) = 3 * \delta_1 + 3 * \delta_2 + 2 * \delta_3$. The different δ ’s are contributed by:

- Item A occurs in both rules and occurs on the same sides of the two rules. So A makes no contribution to the distance. The same happens with C .
- Item D occurs in both rules but occurs on different sides of the two rules. So D makes a contribution of $\delta_2 + \delta_3$ to the distance.
- Item B occurs in one rule and on its left-hand side; it does not occur in the other rule. So B makes a contribution of $\delta_1 + \delta_2$. The same happens with F .
- Item E occurs in one rule, and on its right-hand side; it does not occur in the other rule. So E makes a contribution of $\delta_1 + \delta_3$ to the distance.

Different choice of values for $\delta_1, \delta_2, \delta_3$ can be used to reflect users’ preferences. For most of the paper we will set $\delta_1 = 1, \delta_2 = \frac{n-1}{n^2}$ and $\delta_3 = \frac{1}{n^2}$, where $n = |I|$. However, the approach works for other distance functions.

Having $\delta_1 > \delta_2 > \delta_3$ reflects our belief that the three kinds of item-set differences should contribute differently to the distance: the whole difference $(X_1 Y_1) \ominus (X_2 Y_2)$ is more important than the left-hand side difference $X_1 \ominus X_2$, which in turn is more important than the right-hand side difference $Y_1 \ominus Y_2$.

We set $\delta_2 = \frac{n-1}{n^2}$ and $\delta_3 = \frac{1}{n^2}$ to ensure that (*) rules with identical set of items are closer to each other than to rules with different sets of items. Suppose $R_1 : X_1 \rightarrow Y_1$ and

³ We can define distance functions using the difference of (i) only. However, there can be different rules, namely $A \rightarrow B$ and $B \rightarrow A$, such that the distance between them is 0.

⁴ Given two sets X and Y , $X \ominus Y$ denotes the symmetric difference between X and Y , i.e., $X - Y \cup Y - X$.

$R_2 : X_2 \rightarrow Y_2$ are two rules such that $X_1Y_1 = X_2Y_2$. Then $Dist_{iset}(R_1, R_2) = \delta_2 * |X_1 \ominus X_2| + \delta_3 * |Y_1 \ominus Y_2|$. We wish to ensure that $Dist_{iset}(R_1, R_2) < Dist_{iset}(R_1, R_3)$, for every rule $R_3 : X_3 \rightarrow Y_3$ where $|X_1Y_1 \ominus X_3Y_3| > 0$ (that is R_3 's item set is different from that of R_1 's). We use the following example to further illustrate why (*) holds.

Example 2. Consider the following rules:

$$R_1 : ABC \rightarrow DE \quad R'_2 : DE \rightarrow ABC \quad R'_3 : ABC \rightarrow D.$$

Then $Dist_{iset}(R_1, R'_2) = 0 * \delta_1 + 5 * \delta_2 + 5 * \delta_3 = \frac{5(n-1)}{n^2} + \frac{5}{n^2} = \frac{5}{n} \leq 1 < Dist_{iset}(R_1, R'_3) = 1 * \delta_1 + 1 * \delta_3 = 1 + \frac{1}{n^2} = \frac{n^2+1}{n^2}$. It is easy to see that R'_2 is a rule farthest away from R_1 , among all R_2 containing the same set of items as R_1 . Furthermore, R'_3 is a rule closest to R_1 , among all R_3 containing an item set different from that of R_1 's. Therefore $Dist_{iset}(R_1, R_2) < Dist_{iset}(R_1, R_3)$ for all R_2 and R_3 satisfying the conditions given above.

The $Dist_{iset}$ distance behaves like the usual distances we know:

Proposition 2. *The $Dist_{iset}$ distance is a metric distance over the set of all potential rules. That is, the following properties hold for all rules R_1 , R_2 and R_3 :*

1. $Dist_{iset}(R_1, R_1) = 0$,
2. $Dist_{iset}(R_1, R_2) > 0$ if $R_1 \neq R_2$,
3. $Dist_{iset}(R_1, R_2) = Dist_{iset}(R_2, R_1)$,
4. $Dist_{iset}(R_1, R_3) \leq Dist_{iset}(R_1, R_2) + Dist_{iset}(R_2, R_3)$.

The proofs of (1–3) are easy, and that of (4) can be given by utilizing a vector representation of item sets (after fixing an ordering on elements of I).

As an aside, observe that the potential rule space may not be dense in the sense that every “realizable” distance is the sum of two “realizable” distances. That is, there can be rules R_1 and R_2 such that $Dist_{iset}(R_1, R_2) < Dist_{iset}(R_1, R_3) + Dist_{iset}(R_2, R_3)$ for every rule R_3 ; this happens for $R_1 : AB \rightarrow C$ and $R_2 : AB \rightarrow CD$ and for $R_1 : AB \rightarrow C$ and $R_2 : A \rightarrow BC$. Sometimes, though, a “realizable” distance is the sum of two “realizable” distances: There can be rules R_1 , R_2 and R_3 such that $Dist_{iset}(R_1, R_2) = Dist_{iset}(R_1, R_3) + Dist_{iset}(R_2, R_3)$; for example, $R_1 : AB \rightarrow CDE$, $R_2 : B \rightarrow CD$ and $R_3 : AB \rightarrow CD$.

3.3 Other variants of $Dist_{iset}$

We now suggest some variants of the syntax-based distance by setting $\delta_1, \delta_2, \delta_3$ differently; these can be useful for different user preferences, though not used in the sequel.

If we want to emphasize the changes on the left-hand side of rules, we can set $\delta_1 = 0$, $\delta_2 = 1$, and $\delta_3 = \frac{1}{n+1}$. For any two rules $R_1 : X_1 \rightarrow Y_1$ and $R_2 : X_2 \rightarrow Y_2$ we have

$$Dist_{iset}(R_1, R_2) = |X_1 \ominus X_2| + \frac{1}{n+1}|Y_1 \ominus Y_2|.$$

If we want to emphasize the changes on the right-hand side of rules, we can set $\delta_1 = 0$, $\delta_2 = \frac{1}{n+1}$, and $\delta_3 = 1$. For any two rules $R_1 : X_1 \rightarrow Y_1$ and $R_2 : X_2 \rightarrow Y_2$ we have

$$Dist_{iset}(R_1, R_2) = \frac{1}{n+1}|X_1 \ominus X_2| + |Y_1 \ominus Y_2|.$$

3.4 Distance on other types of rules

We can also define distances for rules of other types. For Datalog (or Horn clauses), we can define distance between two rules in terms of their largest unifiable parts. One can use area of overlap for defining distances between interval-based rules such as $\langle Age : 20..30 \rangle \Rightarrow \langle car : 1..2 \rangle$ ($8\% support, 70\% confidence$).

4 Neighborhoods

In this section, we introduce the notion of neighborhoods of rules. These will be used to define interestingness in the next section.

Definition 3. An r -neighborhood of a rule R_0 ($r > 0$), denoted as $N(R_0, r)$, is the following set

$$\{R : Dist_{iset}(R, R_0) \leq r, R \text{ a potential rule}\}.$$

Although other types of neighborhoods are possible, in this paper we will only be concerned with neighborhoods defined by circles.

One example type of neighborhoods is the 1-neighborhoods such as $N_{R_0,1}$.

Example 3. Suppose $I = \{A, B, C, D, E\}$. Then the 1-neighborhood of the rule $AB \rightarrow CD$ consists of the following rules:

$A \rightarrow BCD$	$B \rightarrow ACD$	$C \rightarrow ABD$
$D \rightarrow ABC$	$AB \rightarrow CD$	$AC \rightarrow BD$
$AD \rightarrow BC$	$BC \rightarrow AD$	$BD \rightarrow AC$
$CD \rightarrow AB$	$ABD \rightarrow C$	$ABC \rightarrow D$
$ACD \rightarrow B$	$BCD \rightarrow A$	

Generally speaking, as a consequence of the way $Dist_{iset}$ is defined, all rules in a 1-neighborhood have the same item set as the center and consequently they all have the same support.

We will also talk about interestingness of collections of rules in terms of their neighborhoods. For example, we can consider the interestingness of the collection of rules in $N_{R_0,1}$ in terms of the set of rules in $N_{R_0,2} - N_{R_0,1} = \{R : 1 < Dist_{iset}(R, R_0) \leq 2, R \text{ a potential rule}\}$. One can also view this set as the union of 1-neighborhoods of all rules whose item set differ from that of the center by exactly one item.

5 Interestingness of rules

In this section we introduce several neighborhood-based interestingness. One of these is in terms of unexpected confidence, and the other is in terms of unexpected density. We then define neighborhood-based interestingness of sets of rules. We also discuss the need for distinguishing the true unexpected changes from the inherent changes in confidence and support due to rule structures.

Similarly one can consider interestingness in terms of unexpected support, especially for collections of rules within some 1-neighborhoods of rules. The details are omitted.

5.1 Interesting rules with unexpected confidence

To capture “unexpected confidence”, we need to introduce two measures of the fluctuation of the confidences of mined rules in a neighborhood: *average confidence* and *standard deviation* of confidence.

Suppose M is a set of mined rules for given minimum support and confidence thresholds min_support and min_confidence , R_0 is a mined rule in M and $r > 0$.

- The *average confidence* of the r -neighborhood of R_0 is defined as the average of the confidences of rules in the set $M \cap N(R_0, r) - \{R_0\}$; we use $\text{avg_conf}(R_0, r)$ to denote this value.
- The *standard deviation* of the r -neighborhood of R_0 is defined as the standard deviation of the confidences of rules in the set $M \cap N(R_0, r) - \{R_0\}$; we will use $\text{std_conf}(R_0, r)$ to denote this value.

When the set $M \cap N(R_0, r) - \{R_0\}$ is empty, we choose to define these two values as zero, although other choices are possible.

Observe that the value $\text{std_conf}(R_0, r)$ gives the average fluctuation of confidences in the r -neighborhood of R_0 .

We choose to identify unexpected confidence of a rule R_0 in its r -neighborhood with the condition that $|\text{conf}(R_0) - \text{avg_conf}(R_0, r)|$ is much larger than $\text{std_conf}(R_0, r)$. This is the basis of our first interestingness in terms of neighborhood-based unexpectedness.

Definition 4. A rule R_0 is said to be *interesting*, of the *unexpected confidence* type, in its r -neighborhood if $||\text{conf}(R_0) - \text{avg_conf}(R_0, r)| - \text{std_conf}(R_0, r)||$ is large⁵; in other words, if the confidence of R_0 deviates from the $\text{avg_conf}(R_0, r)$ much more than the average deviation.

Example 4. Figure 1(a) shows that this definition indeed captures rules with unexpected confidence in a neighborhood. Suppose there are only five mined rules R_1, \dots, R_5 in some r -neighborhood, whose confidences are 0.25, 0.3, 0.75, 0.32, and 0.4 respectively. Then $\text{avg_conf}(R_3, r) = 0.3175$ and $\text{std_conf}(R_3, r) = (((0.25 - 0.3175)^2 + (0.3 - 0.3175)^2 + (0.75 - 0.3175)^2 + (0.32 - 0.3175)^2 + (0.4 - 0.3175)^2)/4)^{\frac{1}{2}} = 0.026$. Thus, $\text{conf}(R_3) - \text{avg_conf}(R_3, r) = 0.4325$, a difference which is about 16 times as large as $\text{std_conf}(R_3, r)$. So, the confidence of R_3 is unexpected in its r -neighborhood.

The value of $||\text{conf}(R_0) - \text{avg_conf}(R_0, r)| - \text{std_conf}(R_0, r)||$ was chosen because it avoids some deficiencies of two other possible alternatives. (i) Defining a rule R_0 as having unexpected confidence if it achieves near maximum difference of confidence between pairs of rules in their r -neighborhood—if there exists $R' \in M \cap N(R_0, r)$ such that $|\text{conf}(R_0) - \text{conf}(R')| \approx \max\{|\text{conf}(R_1) - \text{conf}(R_2)| : R_1, R_2 \in M \cap N(R_0, r)\}$. This suffers from the inability to differentiate between the unexpected minority and the prevailing majority. For example, in Figure 1(a), R_1, R_2, R_3, R_4 (the prevailing majority) will then all be considered as having unexpected confidence. (ii) Defining a rule R_0 as having unexpected confidence if it has no competitors in confidence

⁵ The meaning of large can be specified by a threshold.

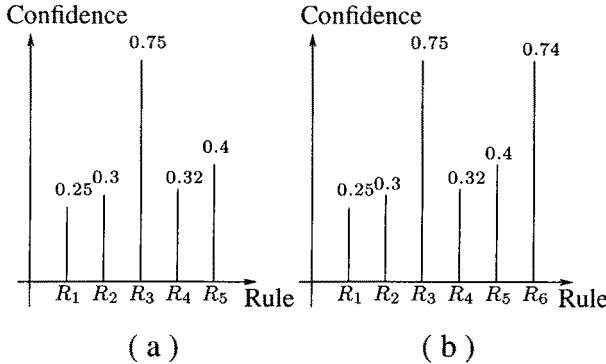


Fig. 1. Unexpected confidence

among rules in a neighborhood—if $\max\{|conf(R_0) - conf(R_1)| : R_1 \in M \cap N(R_0, r)\}$ is large. This definition is not able to capture rules with unexpected confidence when there are two or more rules with outstanding and equal confidence. For example, in Figure 1(b), neither R_3 nor R_6 will then be considered as having unexpected confidence.

The values used in the above definition can be used for ranking the interestingness of rules. For example, the larger the neighborhood the more interesting the rule is, and the larger $| |conf(R_0) - avg_conf(R_0, r)| - std_conf(R_0, r) |$ the more interesting the rule is. This will be the topic of Section 6.

Rules with unexpected confidence can have around the highest confidence among rules in the neighborhood or the lowest. For the latter, however, caution is needed when the confidence of the rule is not sufficiently larger than the minimum confidence threshold: there might be many potential rules in the neighborhood whose confidences are just below the threshold.

5.2 Interesting rules with sparse neighborhoods

A second kind of rules are considered interesting because there are many potential rules in their neighborhoods but there are very few mined rules there.

Definition 5. A rule R_0 is *interesting*, of the *isolated* type, if it has an unexpectedly sparse r -neighborhood: if the number of potential rules in $N(R_0, r)$ is large but the number of mined rules there, i.e. $|M \cap N(R_0, r)|$, is relatively small.

If we call $\frac{|M \cap N(R_0, r)|}{|N(R_0, r)|}$ the *density* of the r -neighborhood of R_0 , then the condition in the above definition can be reworded as “if the number of potential rules in $N(R_0, r)$ is large but the density of the r -neighborhood of R_0 is relatively small.”

Consider Example 3. There are 14 potential rules in the 1-neighborhood of $ABC \rightarrow D$. If $ABD \rightarrow C$ is the only other mined rule in this neighborhood, then the density of this neighborhood is $\frac{2}{14} \approx 14.3\%$. If no rule other than $ABC \rightarrow D$ is mined in this neighborhood, then the density is $\frac{1}{14} \approx 7\%$.

We choose to use the number of potential rules as well as the density $\frac{|M \cap N(R_0, r)|}{|N(R_0, r)|}$ in the definition, because we believe that the first number can help determine the degree of interestingness of isolated rules. For example, as shown in Figure 2, the r_0 -neighborhoods of R_1 and R_2 can both be sparse neighborhoods; the r_0 -neighborhood R_1 is more sparse than that of R_2 if the number of potential rules in $N(R_1, r_0)$ is equal to that in $N(R_2, r_0)$.

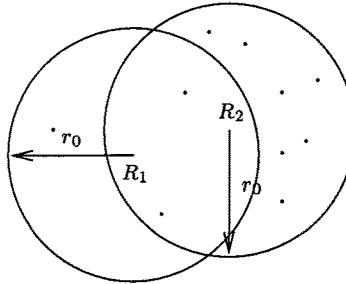


Fig. 2. Sparse neighborhoods

The meaning of “the number of mined rules is relatively small” can be specified by some threshold, either given by the user or calculated from the application (e.g. $\frac{|M|}{N}$, where N is the total number of potential rules).

An isolated rule in a neighborhood may not be a rule with the highest confidence in the same neighborhood. Isolated rules with unexpected confidence are clearly more interesting than isolated rules which do not have unexpected confidence.

Whether a rule is interesting or not depends on the *min_support* and *min_confidence* thresholds. For example, when these thresholds are increased, rules which were not isolated may become isolated.

5.3 Interestingness of collections of rules

Neighborhood-based unexpectedness can also be used to define interestingness of collections of rules. We now briefly describe two such collections, including plateau-like, ridge-like rule groups.

Definition 6. Let M be a set of mined rules, R_0 a rule in M , and $r_0 < r_1$ be two positive numbers. We say the r_0 -neighborhood of R_0 has unexpected confidence in its r_1 -neighborhood if the following hold:

- The standard deviation of confidences of the rules in $M \cap N(R_0, r_0)$ is small.
- The average confidence of rules in $N(R_0, r_0)$ is much larger or smaller than the average confidence of the rules in $M \cap (N(R_0, r_1) - N(R_0, r_0))$.

While the interesting group of rules defined above is about a region with a very regular border, it is also possible to consider regions of rules with irregular borders or even

curves. For example, one can define an *interesting ridge* as a sequence of rules with very high confidence. Such a ridge can be obtained by iteratively choosing rules with the highest confidence, within some small neighborhood of the rule chosen last.

6 Expected change due to rule structure

There are certain expected changes of supports and confidences of rules implied by the structure of the rules. Such expected changes should be taken into account when considering the (un)expectedness of changes.

We first note two such expected changes.

(1) Given two rules R_1 and R_2 , if R_1 's item set is a subset of that of R_2 , then $\text{support}(R_1) \geq \text{support}(R_2)$. For example, $\text{support}(A \rightarrow C) \geq \text{support}(AB \rightarrow C)$.

(2) If R_1 's left-hand side is a subset of that of R_2 and R_1 and R_2 have the same item sets, then $\text{conf}(R_1) \leq \text{conf}(R_2)$. For example, $\text{conf}(A \rightarrow CDB) \leq \text{conf}(AB \rightarrow CD) \leq \text{conf}(ABC \rightarrow D)$; this is because $|m(A)| \geq |m(AB)| \geq |m(ABC)|$.

Such expected changes can happen in a larger scale.

Proposition 7. Let U be a fixed item set and PGroup_j^U be the subset of the potential rules whose item sets are U and which have exactly j items on their left-hand sides. Then the average confidence of rules in PGroup_i^U is less than or equal to the average confidence of rules in PGroup_{i+1}^U for any given set of transactions.

Proof. Let k be the number of elements in U . Let R_1, \dots, R_m be an enumeration of the rules in PGroup_i and R'_1, \dots, R'_n an enumeration of the rules in PGroup_{i+1} . Then $m = \binom{k}{i}$ and $n = \binom{k}{i+1}$; observe that $m = \frac{n(i+1)}{k-i}$.

For each R_j , let S_j be the set of rules in PGroup_{i+1} whose left-hand sides contain the left-hand side of R_j . Then the average of confidences of rules in PGroup_{i+1} is

$$\begin{aligned} \frac{\sum_{j=1}^n \text{conf}(R'_j)}{n} &= \frac{(i+1) \sum_{j=1}^n \text{conf}(R'_j)}{(i+1)n} \\ &= \frac{\sum_{j=1}^m \sum_{R' \in S_j} \text{conf}(R')}{(i+1)n} \geq \frac{(k-i) \sum_{j=1}^m \text{conf}(R_j)}{(i+1)n} = \frac{\sum_{j=1}^m \text{conf}(R_j)}{m} \end{aligned}$$

which is equal to the average of confidences of rules in PGroup_i .

One might tend to believe that such expected changes can happen for the rules satisfying given support and confidence thresholds as well – that the average confidence of rules in MGroup_i^U is less than or equal to the average confidence of rules in MGroup_{i+1}^U , where U is a fixed item set and MGroup_j^U is the subset of the rules (i) satisfying given support and confidence thresholds, (ii) whose item sets are U and (iii) which have exactly j items on their left-hand sides. Interestingly, this is false. Indeed, suppose $U = \{A, B, C\}$ and the following set of transactions is given: the transaction ABC occurs 100 times, the transaction AB 50 times, the transaction AC once, the transaction BC 2 times, the transaction A 9 times, the transaction B 8 times, and the transaction C once. Therefore, $|m(ABC)| = 100$, $|m(AB)| = 150$, $|m(AC)| = 101$, $|m(BC)| = 102$,

$|m(A)| = 160$, $|m(B)| = 160$, and $|m(C)| = 104$. If min_confidence is set as 0.65, then $\text{MGroup}_1 = \{C \rightarrow AB(0.9615)\}$ and $\text{MGroup}_2 = \{AB \rightarrow C(0.6667), AC \rightarrow B(0.9901), BC \rightarrow A(0.9804)\}$. So, the average confidence of Group_1 is greater than that of MGroup_2 .

7 Ranking of interesting rules

The mined rules can be ranked according to their degree of interestingness and then given to the user in the ranked order. Ranking of rules can be done using some primitive characteristics, including support and confidence and some neighborhood-based characteristics. In this section we first list the important primitive characteristics, and then discuss how to rank the rules using these characteristics.

7.1 Primitive characteristics

We associate with each rule some primitive characteristics, which are functions.

Given a rule R and a positive number r , we consider important the following primitive characteristics: $\text{support}(R)$, $\text{conf}(R)$, $\text{avg_conf}(R, r)$, $\text{std_conf}(R, r)$, $\text{avg_supp}(R, r)$, $\text{std_supp}(R, r)$, $\text{potent_size}(R, r)$, $\text{density}(R, r)$. The meaning of the first four items have been explained earlier; $\text{avg_supp}(R, r)$ and $\text{std_supp}(R, r)$ can be defined in manners similar to $\text{avg_conf}(R, r)$ and $\text{std_conf}(R, r)$; $\text{potent_size}(R, r)$ is defined as the number of potential rules of R in its r -neighborhood (which can be calculated from R 's item set, r and I). Recall that $\text{density}(R, r)$ is defined as $\frac{|M \cap N(R_0, r)|}{|N(R_0, r)|}$.

Given a radius r , the primitive characteristics can be obtained quite efficiently using the partitioning method discussed later.

7.2 Ranking

The above primitive characteristics can be combined to form combined characteristics.

One example of such combined characteristics is the function we used earlier in defining interesting rules with unexpected confidence; that function is defined by $f(R, r) = ||\text{conf}(R) - \text{avg_conf}(R, r)| - \text{std_conf}(R, r)||$. Another example of such combined characteristics is $g(R, r)$ defined as $g(R) = 1$ if $|N(R, r)| < 50$ and $g(R, r) = \frac{|M \cap N(R, r)|}{|N(R, r)|}$ otherwise. This function can be viewed as a way of specifying what rules are interesting because they have large and sparse r -neighborhoods.

In general, ranking can be specified by weighted sum of several of such combined characteristics. The ways the combined characteristics are formed and the weighting together specify what are important in an application (or what the users' preferences are).

Ranking can also be augmented by reacting to user feedback. At any moment of time, the user should have looked at the "most interesting rules" produced by the system. After examining a rule R , the user may indicate the relative interestingness of this rule compared with the others, whether he/she is interested in seeing more interesting rules in the neighborhood of the rule just seen, and whether she/he is interested in seeing more rules having the same left-hand side as R , etc. The system can accumulate such feedback and try to adapt to the user's preferences, by perhaps adjusting the functions for the combined characteristics or the weightings. Techniques from neural networks might be helpful here.

8 Implementation issues and a detailed example

In this section, we discuss some implementation issues for finding interesting rules from a set of mined rules. We then give a detailed example to illustrate our ideas.

8.1 First partition then find

To find the interesting rules efficiently from a set M of mined rules, we will first partition M into a number of 1-neighborhoods. We need to have one bucket for each nonempty 1-neighborhood. Recall that rules with the same item sets have identical 1-neighborhood. Then, we can identify buckets with item sets whose corresponding 1-neighborhoods are not empty.

To be able to find a bucket for an item set fast, we have a(n ordered) tree to manage the correspondence between an item set and the physical address of the corresponding bucket. For each node of the tree we have a pair (U, Ad) where U is an item set and Ad is the address of the bucket.

The parent node of a node (U, Ad) is the node whose item set is V such that V is the smallest item set containing U as a subset. (We fix an order on the items. Then we use the lexical order on item sets when we talk about order between item sets.) The tree can be maintained efficiently – we only need to consider insertions.

The partition based on 1-neighborhoods can be directly used to find interesting rules for radius $r \leq 1$. For radius $r > 1$ we can find all $[r]$ -neighborhoods using the tree, by brute force in time $O(p^2)$ where p is the number of nonempty 1-neighborhoods. Observe that the $[r]$ -neighborhoods can be formed by merging the pointer sets for the constituent 1-neighborhoods.

After the proper partitioning is done, we can then find those rules which have unexpected confidence or which are isolated from the proper buckets. For each r and each bucket, this can be done in roughly $O(k^2)$, where k is the number of rules in the bucket.

One might wish to find all radius r and rule R such that R has unexpected confidence (or isolated) in its r -neighborhood. When there are too many of such radius, we can get approximate answers by considering only, say, those radius of the form $i + 0.25 * |R|(\delta_2 + \delta_3)$, $i + 0.5 * |R|(\delta_2 + \delta_3)$, $i + 0.75 * |R|(\delta_2 + \delta_3)$, and $i + |R|(\delta_2 + \delta_3)$, where i is a non negative integer.

8.2 A detailed example

A synthetic example is given below to demonstrate the procedures of finding interesting rules. Suppose the total item set is $I = \{A, B, C, D, E, F\}$. Suppose the thresholds for the confidence and support are set as 0.205 and 0.05, respectively, and the following rules are mined.

$CDE \rightarrow F(.78125)$ $CDF \rightarrow E(.7143)$ $CF \rightarrow E(.5128)$ $EF \rightarrow C(.5)$
 $CEF \rightarrow D(.625)$ $DEF \rightarrow C(.833)$ $E \rightarrow CF(.333)$ $F \rightarrow CE(.2667)$
 $CD \rightarrow EF(.2451)$ $CE \rightarrow DF(.4167)$ $DE \rightarrow F(.3061)$ $DF \rightarrow E(.4)$
 $CF \rightarrow DE(.3205)$ $DE \rightarrow CF(.2551)$ $EF \rightarrow D(.375)$ $D \rightarrow EF(.24)$
 $DF \rightarrow CE(.333)$ $EF \rightarrow CD(.3125)$ $E \rightarrow DF(.25)$ $C \rightarrow D(.51)$
 $E \rightarrow CDF(.2083)$ $CD \rightarrow E(.3137)$ $D \rightarrow C(.816)$ $C \rightarrow E(.3)$
 $CE \rightarrow D(.5333)$ $DE \rightarrow C(.3265)$ $E \rightarrow C(.5)$ $C \rightarrow F(.39)$
 $D \rightarrow CE(.256)$ $E \rightarrow CD(.267)$ $F \rightarrow C(.52)$ $D \rightarrow E(.784)$
 $CD \rightarrow F(.3431)$ $CF \rightarrow D(.4487)$ $E \rightarrow D(.8167)$ $D \rightarrow F(.6)$
 $DF \rightarrow C(.4667)$ $D \rightarrow CF(.28)$ $F \rightarrow D(.5)$ $E \rightarrow F(.67)$
 $F \rightarrow CD(.233)$ $CE \rightarrow F(.6667)$ $F \rightarrow E(.5333)$

The percentage following each rule represents its confidence.

The above 43 mined rules can be partitioned into 11 clusters, each of these being a 1-neighborhood. The biggest one is for the 1-neighborhood centered at the rule of $CDE \rightarrow F$ and contains all rules whose item set is $\{C, E, D, F\}$; the other ten 1-neighborhoods are much smaller and are for rules whose item sets are $\{C, D, E\}$, $\{D, E, F\}$, $\{C, D, F\}$, $\{C, E, F\}$, $\{C, D\}$, $\{C, E\}$, $\{C, F\}$, $\{D, E\}$, $\{D, F\}$, and $\{E, F\}$ respectively.

Observe that the 2-neighborhood of $CDE \rightarrow F$ is the union of the 1-neighborhoods for $\{C, D, E, F\}$, $\{C, D, E\}$, $\{D, E, F\}$, $\{C, D, F\}$, $\{C, E, F\}$, $\{A, C, D, E, F\}$ and $\{B, C, D, E, F\}$.

For the rule $R_1 : CDE \rightarrow F$, the following table shows the neighborhood-based parameters for the different radius values of $\rho_j = i + \frac{|R_1|}{4} * j * (\delta_2 + \delta_3)$, where $i = 0$ and $j = 1, 2, 3, 4$. Observe that R_1 is a relatively interesting rule with unexpected confidence in its ρ_1 -neighborhood where $\rho_1 = 0.17$. There are no isolated rules.

Table 1: The confidence fluctuation in ρ_j -neighborhoods of R_1 , where $\rho_j = \frac{|R_1|}{4} * j * (\delta_2 + \delta_3)$. # stands for $|conf(R_1) - avg_conf(R_1, r)|$, ## for number of potential rules.

r	ρ_1	ρ_2	ρ_3	ρ_4
avg_conf	0.3056	0.4711	0.4264	0.4264
std_conf	0.0786	0.2310	0.2075	0.2075
#	0.4757	0.3102	0.3549	0.3549
$Density$	100%	80%	85%	79%
##	4	10	13	14

9 More discussion on related works

Typical measures of interestingness can be divided into two classes: the objective ones and the subjective ones. The objective ones, such as rule template and rule cover, focus on the importance of rules' structures. The subjective ones, in contrast, depend not only on the structure of a rule and the data, but also on the user who examines the rules.

Two useful subjective interestingnesses are *actionability* and *unexpectedness*. The notion of *actionability* [8, 10] of association rules is based on the usefulness of the rules to user – whether the users can do something because of the rules to their advantage.

Actionability is an important subjective measure of interestingness because users are mostly interested in the knowledge that permits them to do their jobs better by taking some specific actions in response to the newly discovered knowledge. However, it is not an easy matter to decide what rules are actionable; the answer might be obtained only after a period of practical validation.

Unexpectedness can be either subjective or objective. Apparently, if a newly discovered pattern is surprising to the user, then it is certainly interesting. For the subjective ones [6, 11], “surprising” means the discovered knowledge contradicts the user’s beliefs. Therefore, unexpectedness is closely related to beliefs or general impressions. Beliefs can be classified into two types: hard beliefs and soft beliefs. The hard beliefs are the constraints that cannot be changed with new evidence, whereas the soft ones are those that the user is willing to change with new evidence.

The objective unexpectedness can be specified in statistical terms. For example, having support and confidence larger than their corresponding thresholds is one such specification; having a higher chance than that under the independence assumption is another.

Our neighborhood-based interestingness belongs to the class of objective measures of interestingness, because the neighborhoods are determined by the rules’ structures. Clearly, useful interestingness measures should help identify those rules that are surprising to the user. We believe that our neighborhood-based unexpectedness is very useful in this regard, and can be used in complement the other measures.

Rule template was also used to help find interesting rules [5] and it is an objective measure for interestingness. A template is an expression $A_1, \dots, A_k \Rightarrow A_{k+1}$, where, each A_i is either an item name, a class name, or an expression $C+$ or $C*$ (C is a class name). Here $C+$ and $C*$ correspond to one or more and zero or more instances of the class C , respectively.

10 Concluding remarks

We have proposed neighborhood-based unexpectedness as a way of identifying interesting rules. In this approach, the interestingness of a rule depends not only on its own support and confidence but also on the support and confidence of rules in its neighborhood. This idea has not been used by previous interestingness measures, including unexpectedness, actionability, rule cover, rule template.

Neighborhood-based interesting rules proposed in this paper include those with unexpected confidence and those with sparse neighborhood. Similar ideas have been used for identifying interesting sets of rules such as plateaus and ridges. The neighborhood-based parameters have been combined with other parameters to rank the interesting rules. We have also addressed some implementation issues for finding neighborhood-based interesting rules.

We gave a few expected tendencies of changes due to rule structures, which should be taken into account when considering unexpectedness. There might be other similar useful properties.

There are also some issues requiring further research, including: How to use users’ feedback to adjust the functions used in the ranking of interesting rules? How to adjust

the values of δ_1 , δ_2 , and δ_3 to best fit the application? It is also possible to find other types of neighborhood-based interesting rules.

Acknowledgement

Work supported in part by a research grant from the Australian Research Council. The authors benefited from discussions with Ramamohanarao Kotagiri on data mining.

References

1. R. Agrawal, T. Imielinski, and A. Swami. Data mining: a performance perspective. *IEEE Transactions on Knowledge and Data Engineering*, 5(6): 914-925, 1993.
2. R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207-216, Washington, D. C., 1993.
3. M.-S. Chen, J. Han, and P. S. Yu. Data mining: an overview from a database perspective. *IEEE Transactions on Knowledge and Data Engineering*, 8(6): 866-882, 1996.
4. Y. Fu and J. Han, Meta-rule-guided mining of association rules in relational databases. *Proc. 1995 Int'l Workshop. on Knowledge Discovery and Deductive and Object-Oriented Databases (KDOOD'95)*, Singapore, December 1995, pp. 39-46.
5. M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen, and A. I. Verkamo. Finding interesting rules from large sets of discovered association rules. In *Proceedings of the Third International Conference on Information and Knowledge Management*, pages 401-407, Gaithersburg, Maryland, 1994.
6. B. Liu and W. Hsu. Post-analysis of learned rules. In *Proceedings of AAAI*, pages 828-834, 1996.
7. G. Piatetsky-Shapiro. Discovery, analysis, and presentation of strong rules. In *Knowledge Discovery in Databases*, G. Piatetsky-Shapiro and W. J. Frawley, Eds, pages 229-248. AAAI Press/The MIT Press, Menlo Park, CA, 1991.
8. G. Piatetsky-Shapiro and C. J. Matheus. The interestingness of deviations. In *Proceedings of the AAAI-94 Workshop on Knowledge Discovery in Databases*, pages 25-36, 1994.
9. R. Srikant and R. Agrawal. Mining generalized association rules. *IBM Research Report RJ 9963*, 1996.
10. A. Silberschatz and A. Tuzhilin. On subjective measures of interestingness in knowledge discovery. In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, pages 275-281, Montreal, Canada, August, 1995.
11. A. Silberschatz and A. Tuzhilin. What makes patterns interesting in Knowledge discovery systems. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):970-974, December, 1996.
12. H. Toivonen, M. Klemettinen, P. Ronkainen, K. Hätonen and H. Mannila. Pruning and grouping discovered association rules. In *MLnet Workshop on Statistics, Machine Learning, and Discovery in Databases*, Crete, Greece, April 1995.
13. A. Tuzhilin. A pattern discovery algebra. In *Proceedings of the 1997 Workshop on Research Issues in Data Mining and Knowledge Discovery*, 1997.

Point Estimation Using the Kullback-Leibler Loss Function and MML

DAVID L. DOWE

(dld@cs.monash.edu.au)

ROHAN A. BAXTER

(rohan@cs.monash.edu.au)

JONATHAN J. OLIVER

(jono@cs.monash.edu.au)

CHRIS S. WALLACE

(csw@cs.monash.edu.au)

School of Computer Science and Software Engineering

Monash University

Clayton, Victoria 3168, AUSTRALIA

Abstract:

We consider the problem of point estimation of an unknown continuous parameter within a Bayesian setting. Given data-space \mathcal{X} , data $x \in \mathcal{X}$, likelihood function $p(x|\theta)$ and prior $h(\theta)$, we estimate $\hat{\theta}$ using a loss function, $C(\theta, \hat{\theta})$. We are interested in the case where the cost function is unknown at the time of point estimation. Invariance and other properties of the Kullback-Leibler distance lead us to consider it as a ‘reference’ loss function. This involves finding the $\hat{\theta}$ which minimises the expected Kullback-Leibler distance:

$$EKL(\hat{\theta}) = \int_{\theta} \frac{h(\theta)p(x|\theta)}{r(x)} C(\theta, \hat{\theta}) d\theta = \int_{\theta} \frac{h(\theta)p(x|\theta)}{r(x)} \sum_{y \in \mathcal{X}} p(y|\theta) \log \frac{p(y|\theta)}{p(y|\hat{\theta})} d\theta$$

where $r(x) = \int_{\theta} h(\theta)p(x|\theta)d\theta$ is the marginal probability of the data. Rewriting:

$$EKL(\hat{\theta}) = \int_{\theta} \frac{h(\theta)p(x|\theta)}{r(x)} \sum_y p(y|\theta) \log p(y|\hat{\theta}) d\theta - \int_{\theta} \frac{h(\theta)p(x|\theta)}{r(x)} \sum_y p(y|\theta) \log p(y|\theta) d\theta$$

The second term is not dependent on $\hat{\theta}$ and can be omitted from the expression to be minimized. Swapping the order of integration,

$$\begin{aligned} EKL(\hat{\theta}) &= \sum_y \log p(y|\hat{\theta}) [\int_{\theta} \frac{h(\theta)p(x|\theta)}{r(x)} p(y|\theta) d\theta] + \text{constant} \\ &= \sum_y \log p(y|\hat{\theta}) [\int_{\theta} \pi(\theta|x) p(y|\theta) d\theta] + \text{constant} \end{aligned} \quad (1)$$

The term $\pi(\theta|x)$ is the posterior probability of the parameter θ given the data x ; and the term in square brackets is $\text{prob}(y|x)$, the probability of future data y given our current data, x . Given our current data, x , and parameter estimate, $\hat{\theta}$, equation (1) gives the expected log-likelihood of future data, y . We consider a minimum Expected Kullback-Leibler (EKL) estimate, the value of $\hat{\theta}$ which minimises the expression in Equation (1).

Analytical minimisation of this expression (1) is relatively difficult for all but the simplest of cases. However, we can numerically generate simulated future data, y , by generating θ from the posterior (in θ given x) and then generating y from $p(y|\theta)$. The maximum likelihood estimate of this would-be future sample will converge in the limit to our desired minimum EKL estimate.

A related invariant technique is the information-theoretic Minimum Message Length (MML) method [WF87], which entails the minimisation of a two-part message transmitting a hypothesis, θ , and the data, x , in light of θ . The MML estimator is a quadratic approximation to the Strict MML (SMML) estimator [WB75, WF87], which maps partitions of the discrete data-space to estimator values. Like the Kullback-Leibler distance, the SMML and MML estimators are invariant under parameter transformations. The MML es-

timator has been shown to return a relatively small Kullback-Leibler distance from the true distribution for various models, including circular distributions[WD93] and factor analysis.

The MML estimator minimises a two-part message length of transmitting θ and the current data, x , in light of θ . The minimum EKL estimator minimises the expected length of the transmission of future data. We now present an argument as to why these two methods are similar.

First, an approximation to SMML called Fairly Strict MML (FSMML) maps regions from the parameter space to point estimates. These regions, or coding blocks, can then be used to generate synthetic data by convolving $h(\theta)$ with $p(x|\theta)$ over the coding block. Taking the Maximum Likelihood estimate for this synthetic data gives the FSMML estimate, $\hat{\theta}$, for the coding block.

Minimum EKL takes a prior, $h(\theta)$, and data, x , to produce a posterior, $\pi(\theta|x)$. Whether we analytically convolve the posterior with $p(x|\theta)$ or whether we numerically sample a θ from the posterior and then a datum, y , from $p(y|\theta)$, we obtain a population of expected future data. The minimum EKL estimator is the maximum likelihood estimator on the expected future data.

These two methods are very similar, and only differ insofar as the minimum EKL uses the posterior whereas FSMML uses the prior over the coding block. Given that we do not know exactly where our coding block regions might be in FSMML, we can average the location of the coding.

This leads to something very akin to the posterior. It has been shown [Wal96] that the SMML estimator behaves locally very similarly to the posterior, the similarity increasing with the dimension of the parameter space.

We note similarities between MML (or Strict MML) estimators and min EKL estimators for a variety of problems where both estimators are defined. We also note the case of the uniform distribution, for which the Strict MML estimator is defined but the min EKL estimator is not defined.

Finally, in contrast with the successes of the Bayesian SMML and min EKL estimation techniques described above, we conjecture (Dowe, 1997) with slight evidence that no classical estimation technique can always be invariant and statistically consistent while providing internally consistent parameter estimates.

Keywords: Machine Learning, Noise handling, Algorithmic complexity, Bayesian and Statistical Learning Methods, Minimum Message Length, Induction in KDD.

1 Bernoulli sampling

Recalling equation 1, we see by generating θ from the posterior, $\pi(\theta|x)$ in θ and then generating future data, y , from that, the minimum EKL estimator, $\hat{\theta}_{EKL}$, is obtained by minimising the expected K-L distance from the distribution in y .

1.1 Binary Bernoulli sampling

For simple two-state Bernoulli sampling, we have $\theta_2 = 1 - \theta_1$.

$$\begin{aligned} EKL(\hat{\theta}) &= \sum_y \log p(y|\hat{\theta}) \left[\int_{\theta} d\theta \pi(\theta|x)p(y|\theta) \right] + \text{constant} \\ &= \log \hat{\theta} \left[\int_{\theta} d\theta \pi(\theta|x)\theta \right] + \log(1 - \hat{\theta}) \left[\int_{\theta} d\theta \pi(\theta|x)(1 - \theta) \right] \end{aligned} \quad (2)$$

We minimise the EKL distance above by setting the partial derivative of this expression with respect to $\hat{\theta}$ to equal 0.

$$\begin{aligned}\frac{\partial d_{EKL}}{\partial \hat{\theta}} &= \frac{1}{\hat{\theta}} \int_{\theta} d\theta \pi(\theta|x)\theta - \frac{1}{1-\hat{\theta}} \int_{\theta} d\theta \pi(\theta|x)(1-\theta) \\ &= -\frac{1}{1-\hat{\theta}} + \frac{1}{\hat{\theta}(1-\hat{\theta})} \int_{\theta} d\theta \pi(\theta|x)\theta\end{aligned}\quad (3)$$

Hence, $\hat{\theta}_{minEKL} = \int_{\theta} \pi(\theta|x)\theta d\theta$, which is the posterior mean.

1.2 M-state Bernoulli sampling

The 2-state, binary, Bernoulli sampling result above generalises.

Recall that the min EKL estimator can be thought of as the estimator obtained by first generating θ from the posterior, then generating future data, y , given θ , and then minimising the KL distance to this distribution in y .

For each of the M states, $s_1, \dots, s_i, \dots, s_M$, $\int_{\theta} \pi(\theta|x)p(s_i|\theta)d\theta$ equals the posterior mean of the probability of state s_i .

For a prior, $h(\theta_1, \dots, \theta_M) = 1/(M-1)!$, uniform over the $(M-1)$ -dimensional simplex $\theta_i \geq 0$ and $\sum_{i=1}^M \theta_i = 1$, this gives

$$\int_{\theta} \pi(\theta|x)p(s_i|\theta)d\theta = (n_i + 1)/(N + M), \quad (4)$$

where n_i is the number observed in state i and $N = \sum_{i=1}^M s_i$.

The MML estimator for this problem can be shown [WB68] to be $(n + 1/2)/(N + M/2)$.

The uncertainty region of MML estimators is approximately $\sqrt{12/F}$, where F is the expected Fisher information. This corresponds to the size of the MML coding block. Note that this is smaller than the difference between the MML and min EKL estimators, which is of the order of $1/(2N^2)$. Such results will hold for regular priors, although see the counterexample in Section 5 for an irregular Bernoulli prior.

2 Gaussian distribution – $N(\mu, 1^2)$

We assume initially that σ is fixed and equal to unity.

For the min EKL estimator, our marginal distribution in x will be very flat along the range in which we have a uniform prior on μ , and will then tail off at the ends.

With a prior, $h(\mu)$, on μ , uniform over some wide range $[l, u]$, with data $x_1, \dots, x_i, \dots, x_N$, we get that

$$\mu_{MML} = \mu_{MaxLik} = \sum_{i=1}^N x_i/N = \bar{x} \quad (5)$$

The marginal distribution, $r(x)$, on the data can be shown to be very flat and to peak near \bar{x} , somewhere between \bar{x} and $(l+u)/2$.

Since $h(\mu)$ is uniform and $p(x|\theta)$ is Gaussian, this gives us that $\pi(\theta|x)$ is very close to the Gaussian distribution $N(\bar{x}, \sigma^2)$ in the interior of the interval $[l, u]$. Near the values l and u respectively, where $r(x)$ flattens out, $\pi(\theta|x)$ will also flatten out. And, then, for θ outside the range $[l, u]$, since $h(\theta) = 0$, so too, $\pi(\theta|x) = 0$.

2.1 Gaussian distribution – $N(\mu, \sigma^2)$

Using a $1/\sigma$ prior on σ [WD94, WB68], with $s^2 = \sum_{i=1}^N (x_i - \bar{x})^2$,

$$\sigma_{MaxLik} = s^2/N \quad (6)$$

and

$$\sigma_{MML} = s^2/(N-1) \quad (7)$$

For this problem, for which we elect the scale-invariant $1/\sigma$ prior on σ , the minimum EKL estimator will require a numerical solution. It will, however, again be very close to the MML estimator.

3 Probabilistic Prediction – Australian Football League matches

Good [Goo83] and others have advocated the use of probabilistic prediction. We agree with this on the grounds that there are times when we wish to not just to know which is the most probable outcome, but how sure we are of such an outcome.

As Good [Goo83] has observed, the correct reward function to encourage probabilistic prediction is a logarithmic reward function.

The staff of the Department of Computer Science at Monash University, friends and colleagues, have participated in a competition[DFHL96] of making probabilistic predictions of which team would win an Australian Football League (AFL) football match. The reward is logarithmic, and hence one ideally wishes to estimate the posterior mean. Such a competition is also known as fully invested gambling [CT91, Theorem 6.1.2].

The same people also have a more general competition [DFHL96] in which one endeavours to minimise the K-L distance between the probability distribution of the winning margin and a Normal distribution, $N(\mu, \sigma)$.

Probabilistic predictions also permit us to weight over several theories and combine them, as in [Sol64, Cov92].

4 Why the Strict MML and minimum EKL estimators approximately concur

We recall that the MML estimator minimises a two-part message length of transmitting θ and the current data, x , in light of θ , and that the minimum EKL estimator minimises the expected length of the transmission of future data. We now present an argument as to why these two methods are similar when both are defined. (A SMML estimator always exists, but see next section for a case when min EKL is undefined.)

First, as in the abstract, an approximation to SMML called Fairly Strict MML (FSMML) maps regions from the parameter space to point estimates. These regions, or coding blocks, can then be used to generate synthetic data by convolving $h(\theta)$ with $p(x|\theta)$ over the coding block. Taking the Maximum Likelihood estimate for this synthetic data gives the FSMML estimate, $\hat{\theta}$, for the coding block.

Minimum EKL takes a prior, $h(\theta)$, and data, x , to produce a posterior, $\pi(\theta|x)$. Whether we analytically convolve the posterior with $p(x|\theta)$ or whether we numerically sample a θ from the posterior and then a datum, y , from $p(x|\theta)$, we obtain a population of expected future data. The minimum EKL estimator is the maximum likelihood estimator on the expected future data. This knowledge serves us well in comparing and contrasting the estimators.

4.1 Posteriors, coding blocks and false oracles

The two methods FSMML and minEKL are very similar, and only differ insofar as the minimum EKL uses the posterior whereas FSMML uses the prior over the coding block. Given that we do not know exactly where our coding block regions might be in FSMML, we can average the location of the coding.

This leads to something very akin to the posterior. It has been shown [Wal96] that the SMML estimator behaves locally very similarly to the posterior, the similarity increasing and becoming more global as the dimension of the parameter space increases.

Also, the posterior probability of the SMML theory is approximately greater than or equal to $(\pi e/6)^{-d/2}$, where d is the dimensionality of the problem under consideration.

4.2 Invariance

Since the Kullback-Leibler distance is invariant under re-parameterisation, it is quite clear that the min EKL estimator remains invariant under re-parameterisation. Both the Strict MML estimator [WF87] and its quadratic approximation, the MML estimator [WF87, WB68] are invariant under re-parameterisation.

4.3 Stability of estimators to local changes in the prior

Another way of thinking about this is that, even if the prior is badly behaved, the prior is smoothed out with the likelihood to give a posterior for min EKL from which a parameter value is sampled. Another sampling is taken when the future data is sampled from a distribution according to the sampled parameter value. Hence, a local change in the prior should not affect the min EKL estimator very greatly.

The SMML estimator will be affected more (see first counter-example below in Section 5.1), but will still typically not change greatly. This is because the SMML estimator maps data onto coding blocks whose prior probability is the sum of the marginal probabilities of the mapped data. The integration in obtaining the marginal probability of the data will smooth out some non-uniform regions in the prior.

This comment is not in general true for the MML estimator, whose quadratic derivation assumes a well-behaved (locally flat) prior distribution. The MML estimator, $\hat{\theta}_{MML}$ of θ is [WF87] that value of θ maximising $\frac{h(\theta)p(x|\theta)}{\sqrt{F}}$, where F is the expected Fisher information.

(See also the Bernoulli counter-example in Section 5.)

4.4 Shifting coding blocks and potential differences in the estimators

In the Bernoulli example in Section 1, we mention that we expect to find the min EKL estimator within the uncertainty region (or coding block) of the MML estimator. The MML estimator can be a continuous function of the sufficient statistics, whereas the SMML estimator is discrete. Where the observed data ended up being border-line between two neighbouring SMML coding blocks, it is conceivable that the min EKL estimator might take a value corresponding to the range of the neighbouring coding block.

4.5 MML, SMML and min EKL estimates for nested model spaces

Consider a problem of nested model spaces, such as (e.g.) trying to fit polynomials of arbitrary degree to some data, trying to fit a boolean decision tree (or classification tree) or trying to fit a mixture model of some unlimited number of Gaussian components to some data. Let us focus on the polynomial example.

The MML estimator will typically choose a polynomial of degree substantially less than the number of data, N . Indeed, so will most other published methods.

However, unless the observed data in the problem admits some special symmetries, both the min EKL estimator and the SMML estimator will typically select a polynomial of degree $N - 1$. The higher order coefficients will most likely be small in magnitude, but they will at least as likely be non-zero. In the case of the min EKL estimator, this should be fairly

clear. In the case of the SMML estimator, recall that each coding block is mapped onto by some countable subset of the (potential) data-space. Just as the min EKL estimator is obtained by obtaining the Maximum Likelihood estimate to pseudo-future data generated from parameters from the posterior, we recall that the SMML estimator is obtained by obtaining the Maximum Likelihood estimate to pseudo-future data generated from parameter values *within the coding block*.

Both of these methods can be expected to give a polynomial of degree $N - 1$.

One comment can be made here concerning the difference between the SMML and min EKL estimators in this case of nested models.

We recall results of Solomonoff [Sol64] and Cover [Cov92] on optimal ways to combine theories predictively. The SMML theory will be (by definition) the theory of the highest posterior probability. In obtaining the minimum EKL estimate, as much as possible, we wish to weight the various theories according to their message lengths. The SMML polynomial will have the largest coefficient amongst all polynomials contributing to the min EKL polynomial.

A similar remark applies regarding combining and averaging decision trees, and highlights the predictive gains to be obtained from judicious over-fitting.

Having highlighted the similarities between SMML (or MML) and min EKL estimation and hinted at some differences, we now highlight some differences.

5 Two counter-examples where MML and min EKL differ

Having argued above as to why we expect MML and min EKL to have much in common, the first example below shows how an irregular prior can lead to a substantial difference between the MML and min EKL estimators.

The SMML estimator is always defined. Our second example gives a case in which the min EKL estimator is not defined.

5.1 Counter-example 1 – an irregular Bernoulli prior

Consider, for sake of argument, a football game subject to some highly variable but quite predictable weather. As the opposing captains await the toss of the unbiased coin, they are both aware that the team kicking with the strong wind in the first quarter is likely to amass a huge lead, after which the wind is expected to reliably change direction again favouring the same team. This will then be followed by heavy rain and hail, substantially restricting further scoring. The teams are otherwise quite evenly matched, but the game will almost literally come down to a coin toss. Both captains have a 50% chance of winning the coin toss, after which the prior probability of victory is uniform between 0.9 and 1.0 for the team initially kicking with the wind.

Prior to the coin toss, the prior distribution on the outcome is uniform in the range $[0, 0.1] \cup [0.9, 1.0]$, and 0 elsewhere.

We turn on the radio some time after the match has begun and, upon hearing that the scores are tied at one goal apiece but without knowing which captain won the coin toss, we wish to estimate the probability of victory of our favourite team. We recall from Section 1 that the min EKL estimator is the posterior mean. This is $0.5 \times (0.05 + 0.95) = 0.5$.

However, the MML and SMML estimates must clearly lie within the support of the prior, i.e., within the range $[0, 0.1] \cup [0.9, 1.0]$.

For the problem as posed, either 0.1 or 0.9 is equally good as the MML estimator. Were we to vary the problem slightly and have the home team leading by 4 goals to 3, we could get a definite choice of 0.9 as the estimator while the min EKL estimator stays near 0.5.

The reader is invited to provide an alternative example.

5.2 Counter-example 2 – the uniform distribution

Consider a probability distribution which is the characteristic function of the interval $[s, s+1]$ for some s .

If we sample N data points from within the interval $[s, s+1]$, we have two sufficient statistics, $a = \min_i x_i$ and $b = \max_i x_i$. As long as $s \leq a \leq b \leq s+1$, the likelihood function, $p(s|x_1, \dots, x_N)$ is constant. Assume that $b < a+1$.

For s outside the range $(b-1, a]$, the likelihood is 0.

Assuming s to be equally likely a priori between $(b-1)$ and a or on some wider such range, we get that, a posteriori, s has probability $1/(a+1-b)$ uniform over the range $(b-1, a]$ and 0 elsewhere.

Hence, we can generate data by sampling s_j from $(b-1, a]$ and then generating future data y_j uniformly from the interval $[s_j, s_j + 1]$.

The p.d.f. will take non-zero values in the region from $(b-1)$ to $(a+1)$, a total range of $a - b + 2$, which is greater than unity. The expected K-L distance from this distribution to a characteristic function of any interval $[\hat{s}, \hat{s}+1]$, whose support is only unity, will be infinite. This result holds asymptotically.

The Strict MML estimator for this problem is described in [WB75]. In short, since the SMML estimator only needs to transmit a two-part message stating \hat{s} followed by the observed data given \hat{s} , the SMML estimator clearly exists.

6 Consistency and invariance in Bayesian inference

Although Maximum Likelihood inference is invariant under 1-to-1 re-parameterisations, it is known to be statistically inconsistent for the Neyman-Scott problem[NS48, DW97], factor analysis[Wal95, WF92] and even mixture modelling. While one can marginalise Maximum Likelihood estimates to obtain consistency (as once suggested by R. A. Fisher[Fis53]), doing this for example in factor analysis results in estimates of the factor scores which do not concur with the estimates of the factor loads. Similarly, although marginal Maximum Likelihood does give a statistically consistent estimate of σ for the Neyman-Scott problem, such an estimate does not concur with the estimates of the μ_i : i.e., the estimates of the μ_i are not internally consistent with the marginal ML estimate of σ .

Contrasting this with the invariance[WB75, WF87] and statistical consistency[Wal96, BC91] of SMML estimation and the internal consistency of MML and SMML estimation, we are led to make the following conjecture.

Conjecture (Dowe, 1997) :

Any estimation technique which is always both invariant and statistically consistent while always providing internally consistent (concurring) parameter estimates must necessarily be Bayesian.

With regard to the conjecture above, we note in general that Maximum Likelihood has difficulties when the number of parameters to be estimated grows as a proportion of the amount of data. We also note in passing small samples biases of Maximum Likelihood in estimating the scale parameters in the von Mises[WD93] and the Gaussian distributions. Regarding Bayesian estimation, we note elsewhere[DOW96, page 217] our preference for subjective Bayesian priors over “mathematically convenient” priors.

7 Conclusion

The minimum expected Kullback-Leibler distance (min EKL) estimator and the Strict MML (or MML) estimator are two invariant Bayesian estimation techniques. SMML has been studied fairly thoroughly, and is known also to be consistent and efficient.

Various theoretical and some empirical similarities have been shown between SMML and the relatively unstudied min EKL estimator.

The uniform distribution, whose support is less than the parameter space, gives an example where the min EKL estimator is not defined.

Where min EKL is defined, we typically find that it is close to the MML estimator.

Furthermore, while min EKL (where defined) and SMML are both invariant and give rise to internally consistent (concurring) estimates, we believe that min EKL (where defined) shares with SMML the property of also always giving statistically consistent estimates. We conjecture above with slight supporting evidence that, in general, no classical estimator can simultaneously have all these properties.

References

- [BC91] A.R. Barron and T.M. Cover. Minimum complexity density estimation. *IEEE Transactions on Information Theory*, 37:1034–1054, 1991.
- [Cov92] T.M. Cover. *IEEE Information Theory Newsletter*, pages 1–6, 1992.
- [CT91] T.M. Cover and Joy A. Thomas. *Elements of Information Theory*. John Wiley and Sons, Inc., New York, 1991.
- [DFHL96] D.L. Dowe, G.E. Farr, A.J. Hurst, and K.L. Lentin. Information-theoretic football tipping. In N. de Mestre, editor, *Proc. 3rd Conference on Mathematics and Computers in Sport*, pages 233–241, Bond University, Qld., Australia, 1996.
- [DOW96] D.L. Dowe, J.J. Oliver, and C.S. Wallace. MML estimation of the parameters of the spherical Fisher distribution. In A. et al. Sharma, editor, *Proc. 7th Conf. Algorithmic Learning Theory (ALT'96), LNAI 1160*, pages 213–227. Sydney, Australia, October 1996.
- [DW97] D.L. Dowe and C.S. Wallace. Resolving the Neyman-Scott problem by Minimum Message Length. In *Proc. Computing Science and Statistics - 28th Symposium on the interface*, volume 28, pages 614–618, 1997.
- [Fis53] R.A. Fisher. Dispersion on a sphere. *Journal of the Royal Statistical Society (Series A)*, 217:295–305, 1953.
- [Goo83] I.J. Good. Explicativity, corroboration, and the relative odds of hypotheses. In *Good thinking— The Foundations of Probability and its applications*. University of Minnesota Press, Minneapolis, MN, 1983.
- [NS48] J. Neyman and E.L. Scott. Consistent estimates based on partially consistent observations. *Econometrika*, 16:1–32, 1948.
- [Sol64] R.J. Solomonoff. A formal theory of inductive inference. *Information and Control*, 7:1–22, 224–254, 1964.
- [Wal95] C.S. Wallace. Multiple Factor Analysis by MML Estimation. Technical Report 95/218, Dept. of Computer Science, Monash University, Clayton, Victoria 3168, Australia, 1995. submitted to J. Multiv. Analysis.

- [Wal96] C.S. Wallace. False Oracles and SMML Estimators. In D.L. Dowe, K.B. Korb, and J.J. Oliver, editors, *Proceedings of the Information, Statistics and Induction in Science (ISIS) Conference*, pages 304–316, Melbourne, Australia. August 1996. World Scientific. Was Tech Rept 89/128. Dept. Comp. Sci., Monash Univ., Australia, June 1989.
- [WB68] C.S. Wallace and D.M. Boulton. An information measure for classification. *Computer Journal*, 11:185–194, 1968.
- [WB75] C.S. Wallace and D.M. Boulton. An invariant Bayes method for point estimation. *Classification Society Bulletin*, 3(3):11–34, 1975.
- [WD93] C.S. Wallace and D.L. Dowe. MML estimation of the von Mises concentration parameter. Tech Rept TR 93/193, Dept. of Comp. Sci., Monash Univ., Clayton 3168, Australia, 1993. prov. accepted, Aust. J. Stat.
- [WD94] C.S. Wallace and D.L. Dowe. Intrinsic classification by MML – the Snob program. In C. Zhang and et al., editors, *Proc. 7th Australian Joint Conf. on Artif. Intelligence*, pages 37–44. World Scientific, Singapore, 1994.
- [WF87] C.S. Wallace and P.R. Freeman. Estimation and inference by compact coding. *J. Royal Statistical Society (Series B)*, 49:240–252, 1987.
- [WF92] C.S. Wallace and P.R. Freeman. Single factor analysis by MML estimation. *Journal of the Royal Statistical Society (Series B)*, 54:195–209, 1992.

Single Factor Analysis in MML Mixture Modelling

Russell T. Edwards^{1,2} and David L. Dowe²

¹ Supercomputing and Astrophysics Research Group, Swinburne University of Technology, John St, Hawthorn, Victoria 3122, Australia
`redwards@pulsar.physics.swin.edu.au`,

² School of Computer Science and Software Engineering, Monash University, Wellington Rd, Clayton, Victoria 3168, Australia
`dld@cs.monash.edu.au`

Abstract. Mixture modelling concerns the unsupervised discovery of clusters within data. Most current clustering algorithms assume that variables within classes are uncorrelated. We present a method for producing and evaluating models which account for inter-attribute correlation within classes with a single Gaussian linear factor. The method used is Minimum Message Length (MML), an invariant, information-theoretic Bayesian hypothesis evaluation criterion. Our work extends and unifies that of Wallace and Boulton (1968) and Wallace and Freeman (1992), concerned respectively with MML mixture modelling and MML single factor analysis. Results on simulated data are comparable to those of Wallace and Freeman (1992), outperforming Maximum Likelihood. We include an application of mixture modelling with single factors on spectral data from the Infrared Astronomical Satellite. Our model shows fewer unnecessary classes than that produced by AutoClass (Goebel et. al. 1989) due to the use of factors in modelling correlation.

Keywords: Minimum Message Length, MML, Statistical and Machine Learning, Noise Handling, Induction in KDD.

1 Introduction

This paper introduces a method for inferring models of data based on the combination of two kinds of model frameworks: mixture models [13, 16, 14, 18] and factor models [7, 20, 15]. Mixture modelling, variously also known as a form of clustering, intrinsic classification, and numerical taxonomy, is the modelling of a statistical distribution by a mixture of other distributions, known as components or classes. Mixture modelling is prone to overfitting since the best fit to any body of data, in terms of maximum likelihood, is typically achieved by having many classes, each fitting a small number things ¹. The Minimum Message Length (MML) principle, presented by Wallace and Boulton [16], is a Bayesian method

¹ where a ‘thing’ is a datum of observed attribute values

which penalises overly complex hypotheses, and for this reason it has been used for the evaluation of mixture model hypotheses [16, 14, 18]².

Factor analysis concerns the modelling of inter-attribute correlation by the assertion of the existence of some attribute of things which was not measured, but which has an effect on the attributes that were observed. The MML estimator for factors [20, 15] produces results that are superior on average to maximum likelihood estimators.

Real-world data often exhibits correlation structure which is well suited to modelling by a mixture of distributions with factors. We have developed an MML method for the estimation of the parameters and structure of this class of models. We present an analysis of the results of this estimator on simulated data and on spectral data from the Infrared Astronomical Satellite (IRAS) [12], which shows the value of the inclusion of factors in mixture modelling. Spectral data contains a large amount of correlation, much of which corresponds to the variation in strength of continuous parameters of sources, such as their temperature or the abundance of certain ions and molecules. This kind of variation may be handled by mixture models (Goebel et. al. [6] have published such a model for the IRAS data), but we believe it is better modelled by mixtures of factors, resulting in a much reduced number of classes.

2 Parameter Estimation by Minimum Message Length

The minimum message length (MML) principle as stated by Wallace and Boulton [16] asserts that the “best” hypothesis about data is that which minimizes the length of a two part message conveying the hypothesis and encoding the data given this hypothesis. Elementary information theory results tell us that an event of probability p may be encoded in $-\log_2 p$ bits. Therefore, assuming a prior distribution on hypotheses, we may encode our hypothesis H in length $-\log_2 \Pr(H)$ bits, or $-\log_e \Pr(H)$ nits, a unit of convenience since we are often taking the logarithm of exponentials of base e . Given the hypothesis, which somehow conveys a probability distribution for the data (by, for example, encoding the mean and standard deviation for a Normal distribution), the data are encoded in $-\log \Pr(D|H)$ nits. Hence, the entire message is conveyed in $-\log \Pr(H) - \log \Pr(D|H) = -\log \Pr(H) \cdot \Pr(D|H)$ nits, and the MML estimate minimizes this quantity, or equivalently, maximizes $\Pr(H) \cdot \Pr(D|H)$. By Bayes’ rule this corresponds to maximizing $\Pr(D) \cdot \Pr(H|D)$, and since $\Pr(D)$ is independent of H , the MML estimate therefore maximizes the Bayesian posterior probability $\Pr(H|D)$.

Unlike the usual Bayesian method of maximizing the posterior density, MML works with probability masses by recognizing that all data is necessarily measured to a finite precision, and that point estimates should also be recorded to a finite precision. Thus the precision of all distribution parameters, as well as their expected values under quantization are estimated. The precision of distribution

² More information regarding mixture modelling with MML and other methods is available at <http://www.cs.monash.edu/~dld/mixture.page.html>.

parameters is determined as a trade-off between the cost of encoding with extra precision, and the cost of encoding the data with a sub-optimal hypothesis due to rounding. For continuous-valued parameters, with the optimum quantizing volume (to a second order quadratic Taylor expansion approximation), the hypothesis is encoded in $-\log \frac{h(\mathbf{z})}{\sqrt{k_D^D F(\mathbf{z})}}$ nits, where $h(\mathbf{z})$ is the prior density at the D -dimensional parameter vector estimate \mathbf{z} , k_D is the optimal D -dimensional lattice constant and $F(\mathbf{z})$ is the determinant of the matrix of expected second partial derivatives of the negative log-likelihood with respect to the elements of \mathbf{z} (known as the Fisher information). Given the hypothesis \mathbf{z} encoded to this precision, the second part of the message incurs an overhead due to rounding of the parameters of $D/2$ nits, giving a length of $-\log \Pr(D|\mathbf{z}) + D/2$. The MML estimate of \mathbf{z} is that which minimizes the length of this two part message. Unlike the Bayesian maximum a posteriori estimate which maximizes the posterior density, the MML estimator is invariant under re-parameterisation. For a more detailed treatment see [19] or [17]. For estimators of the parameters of specific probability distributions, see [18].

3 Mixture Modelling by Minimum Message Length

3.1 Fundamentals

The original application of MML by Wallace and Boulton [16] was in mixture modelling. Recall that MML involves the evaluation of the length of a hypothetical message describing a hypothesis and encoding the data based on this hypothesis. For mixture models, we assume a message composed of the following parts:

- 1a. The number of components. (All numbers are considered equally likely a priori, although this could easily be modified.)
- 1b. The relative abundance of each component. (Creating names or labels for each component of length $-\log_2$ of the relative abundance, via a Huffman code, gives us a way of referring to components later when, e.g., we wish to say which component a particular data thing belongs to.)
- 1c. For each component, the distribution parameters of the component
- 1d. For each thing, the component to which it is estimated to belong. (This can be done using the Huffman code referred to in 1b above.)
2. The attribute values of each thing in turn, encoded using parameters of the hypothesis.

Given an assignment of things to classes, this message format allows the independent estimation of the distribution parameters of each class and the relative abundances of each class to minimize parts 1a-c of the message. Likewise, given distribution parameters and relative abundances, it allows the independent assignment of each thing to a class to minimize parts 1d and 2 of the message.

3.2 Snob and the EM Algorithm “Adjust Cycle”

The Snob [16, 14, 18] program for MML mixture modelling uses a version of the Expectation Minimization (EM) algorithm [10] to find a model with locally minimum message length. Given some initial number of classes and assignment of things to classes, the *first step* is to estimate the distribution parameters which minimize the length of the parts of the hypothesis and the data which pertain to each attribute of each class. Given these estimates for the parameters of each class, the *second step* is to re-assign things to classes in a manner which minimizes the combined length of parts 1d and 2. These two steps are repeated until the message length has reached convergence, at which point a local minimum of message length must have been found.

The topology of the mixture model is evolved by considering at each step of iteration the effect of either promoting the iterated *sub-classes* of a class, or *combining* two classes into one, and performing the alteration if it would decrease the message length.

3.3 Partial Assignment

Part 1d of the message described in the previous section implicitly restricts us to hypotheses, H , which assert with 100% definiteness which component each thing belongs to. Given that the population that we might encounter could consist of two different but highly over-lapping distributions, forcing us to state definitely which component each thing belongs to is bound to cause us to mis-classify outliers from one distribution as belonging to another. In the case of two overlapping (but distinguishable) 1-dimensional Normal distributions, this would cause us to over-estimate the difference in the component means and under-estimate the component standard deviations. Since what we seek is a message which enables us to encode the attribute values of each thing as concisely as possible, we note that a shorter message can be obtained by a probabilistic (or partial) assignment of things to components. The reason for this is that[14, p77] if $p(j, x), j = 1, \dots, J$, is the probability of component j generating datum x , then the total assignment of x to its best component results in a message length of $-\log(\max_j p(j, x))$ to encode x whereas, letting $P(x) = \sum_j p(j, x)$, a partial assignment of x having probability $p(j, x)/P(x)$ of being in component j results in a shorter message length of $-\log(P(x))$ to encode x . As shown in [14, p77], this shorter length is achievable by a message which asserts definite membership of each thing by use of a special coding trick.

4 Single Factor Analysis by Minimum Message Length

Classification is a way of modelling inter-attribute correlation. Whilst mixtures of uncorrelated multivariate distributions are able to model any form of inter-attribute correlation, they do so at a cost if the correlation does not arise due to actual homogeneous unknown sub-populations in the things to be modelled.

Typically in real-world data we might expect to find correlation structure of a continuous kind rather than just the disjoint style of mixture models. One way of modelling this is to hypothesize the existence of some continuous-valued attribute of things, which was not measured, and which represents a common “factor” affecting the attribute values that *were* measured. Hence, the attribute values of a thing, \mathbf{x}_n where $x_{nk} \in \mathbb{R}$ are modelled by

$$x_{nk} = \mu_k + v_n a_k + \sigma_k r_{nk} \quad (1)$$

and v_n and r_{nk} are independently distributed from $N(0, 1)$ for $1 \leq n \leq N$ and $1 \leq k \leq K$, where N is the number of things and K is the number of attributes per thing. The r_{nk} term represents the usual model of uncorrelated Gaussian variance, whilst $v_n a_k$ models the effect of the common factor: v_n is the *factor score* of thing n and a_k is the *factor load* for attribute k . Thus the effect of the factor on the value of attribute k of thing n is characterised by the factor score a_k with which it affects the k th attribute values of all things, and the factor load v_n with which all the attribute values of thing n are affected by the factor.

One example of modelling by factors is the concept of the Intelligence Quotient (IQ). IQs by definition are Normally distributed among the world’s population (or, at least, among the people who have had IQ tests!) with a standard mean and variance. A person’s IQ is estimated from their results on one or more IQ tests, for it is assumed that intelligence is a monotonic function of score on such tests. In terms of factors, a person’s IQ is the factor score, and the factor loads embody what each intelligence test tells us about IQ, whilst the person’s results at different tests would comprise their attributes. A high factor score would indicate high IQ, and one would expect that people with high IQ would perform well on most intelligence tests. A large positive factor load would indicate a test which is highly sensitive to intelligence in terms of the variation in its results, whilst a zero factor load would indicate a test which produces results that are totally uncorrelated to the intelligence of the person taking it.

For this kind of data, a mixture model must become overly complex in order to achieve a good fit — the usual result is that for each true class in the data (there may be just one), a mixture model produces several components in order to cover it fully. This effect is illustrated in figure 1, which depicts some two-dimensional data generated with a single class containing a very strong factor ($|\mathbf{a}|/|\sigma| = 5$), and the probability density assigned to the data space by the MML mixture model of this data. The mean of the data appears around the middle of the plane, with a factor load vector of the same sign and (as plotted) equal magnitude in both attributes, as evidenced by the spread of data along the diagonal. The sign of \mathbf{a} is arbitrary, but assuming a_1 and a_2 are positive, data points toward the upper right side of the plane would have large positive factor scores. Note that the optimal mixture model of this data produced three classes dotted along the factor to account for the correlation, where clearly there is only one intrinsic population in the data. This process of assigning multiple classes to account for continuous variation is similar to the way some human-designed classification schemes cope with correlation: a number of sub-classes

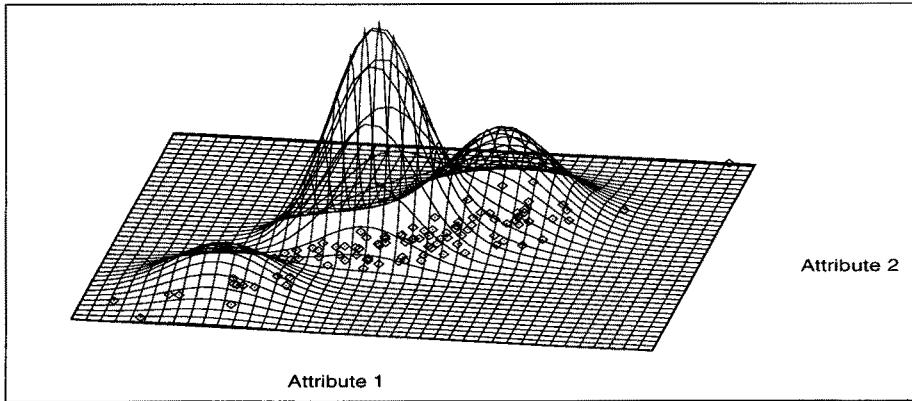


Fig. 1. Pseudo-randomly generated data with a strong Gaussian factor. The points (which are on the $z = 0$ plane) show the data points, whilst the surface shows the probability density over attribute values, assigned by the MML mixture of three Gaussian classes.

are produced for every real class, with each sub-class (e.g. low, medium, high) spanning a certain range of data corresponding to different strengths of the factor (factor scores). Since we are interested in inferring something about the true class structure of the data (which in this case means only a single class, not three), it is better to model linear correlation directly.

An MML estimator for single factor models of the form in equation 1 was developed by Wallace and Freeman [20]. This estimator allows simultaneous estimation of the factor loads and scores, unlike maximum likelihood estimators, which cannot do so in a consistent manner. In addition, where there was evidence for a common factor this estimator is more accurate than the Maximum Likelihood estimators, and where there is insufficient evidence for a factor, the uncorrelated multivariate Gaussian model is reverted to by virtue of its smaller message length. Wallace [15] has also derived an MML estimator for models involving multiple factors.

5 MML Mixture Modelling with Single Factors

It is the nature of many mixture modelling applications that the data would be well modelled by a set of classes with the presence of factors with each class, or perhaps, shared across classes. We have developed a method for MML estimation of such a model.

Recall the mixture model message format described in section 3.1. The amendment of this format to allow factor models is conceptually simple: for part 1c we first encode the distribution parameters of any attributes not to be modelled as Gaussian (for example, discrete values [16], angles [17], non-negative reals). We then specify whether or not the remaining attributes are modelled with a factor. If the two choices are equally likely a priori, this specification is of

constant message length and may be ignored for the purposes of model discrimination within this framework. If a factor is not used, the means and standard deviations are encoded as usual, otherwise the factor parameters, consisting of the means, standard deviations, loads and scores, are encoded. Other parts of the message remain in the same format as before.

5.1 Modifications for Mixture Models Incorporating Factors

The search strategy used by Snob and described in section 3.1 works on the assumption that step two of the adjust cycle (section 3.2), the re-assignment of things to classes, will not affect the length of parts 1a-1c of the message. This appears not to be strictly true since in order to be able to send the second part of the message in a length equal to the negative log likelihood plus the small constant rounding costs (section 2), the estimates must be the MML estimates, encoded to the correct precision. Just as in section 2 it was assumed that the hypothesis cost varies little with the rounding of estimates, here it is assumed that the length of the hypothesis is a slowly varying function of the data and thus that any corrections that ought to be made to allow costing of the second part of the message as described, would have negligible effect on the length of parts 1a-1c. However, a small change in the number of things to be modelled by a factor (as occurs in step two of the adjust cycle) results in a large change in the cost of the hypothesis, as there will be a different (possibly larger or smaller) set of factor scores to encode.

When re-assigning a thing to a different class, we must do so based on a knowledge of not only how this will affect the length of parts 1d and 2 of the message, but seemingly also on how this will affect the length of the new part 1c which must arise as a result of re-assignment. Mainly, we must take into account the difference between the cost of describing the factor score of this thing in its new class compared to its old class, which may be large, especially if one of the classes is not modelled with a factor, meaning no factor score need be specified. If this cost is not properly taken into account, the message length can actually increase over successive iterations of the adjust cycle.

The Fisher information matrix for factor models is not diagonal in the elements involving the factor scores, which means that in transformation of the parameter space to allow optimal independently quantised coding [19], the factor scores are not separable from some other parameters. This means that we cannot simply attribute a single cost for the specification of a factor score, for use in re-assignment as described above. Recall from section 3.2 that step two of the adjust cycle for mixtures of uncorrelated distributions, re-assignment of things to classes in a manner which minimizes the lengths of parts 1d and 2 (and now, 1c) may be done optimally by considering the class assignment of each thing independently from that of every other thing. This is not true for mixtures of factors, because of the impossibility of apportioning the class-assignment-dependent parts of 1c (the specification of factor scores) to the culprit things. As a result of this, it seems the only way to perform step 2 of the adjust cycle optimally whilst taking into account the effects re-assignment has on the length of part 1c is to

find an overall assignment of all things to classes at once which minimizes the new lengths of parts 1c, 1d and 2, given that all estimates in 1c are to remain the same, apart from the factor scores.

As a result of these implications of the factor model, optimal re-assignment in step 2 of the adjust cycle appears to become an infeasible combinatorial search problem of exponential complexity, as compared to the usual Snob adjust cycle with its independent assignment of things in linear time. Considering the intractability of such a search, we have devised an iterative hill-climbing scheme. This scheme assigns one thing at a time by considering the effect the assignment would have on parts 1c, 1d and 2. This process is applied repetitively to random things, meaning that the assignment of things is continually updated in random order (to avoid bias), until a locally minimal assignment is found. For all moderate to large datasets this process becomes too slow and instead each thing is re-assigned only once, which is equivalent to considering what effect the assignment would have if the thing was the only thing whose assignment changes. This is not optimal except when the Snob adjust cycle has converged, but in practice it appears to always reduce the message length.

5.2 Partial Assignment with Factors

The method of re-assignment described in the previous section assumes total assignment, however it is desirable to use partial assignment to produce a consistent estimator (see section 3.3). The coding trick for partial assignment does not appear to be compatible with the scheme described in the previous section, for it requires that we be able to cost, or estimate the cost, of the parts of 1c, 1d and 2 that are attributable to each thing. In the previous section we calculated only the overall change in cost of 1c for a simultaneous assignment of all things to classes.

As noted in section 5.1, there appears to be no way of separating the costs of individual factor scores from the other factor parameters, and hence there is no way to distribute the cost as this scheme would require. Wallace [15] uses a polar representation for the factor score vectors in models involving multiple factors, which may allow independent coding.

5.3 Relationship to Previous Work

Hinton et. al [8] present an interesting method for fitting mixtures of single factors, based on a type of neural network called an “autoencoder” [9]. The network attempts to duplicate the input data at its output. A model is fitted essentially by minimizing the description length of a two-part message consisting of the factor scores and the differences between the input and output. Since it ignores the cost of all other parameters (e.g. the distribution parameters of each class, which should penalize mixtures of too many components), and encodes the output errors according to a fixed, empirical prior, this method appears equivalent to Maximum Likelihood, marginalized over factor scores [20] and assuming a prior value for σ . Maximum Likelihood is known to over-fit in problems where

the number of parameters to be estimated grows rapidly [11, 5, 20, 15], in fact it is the conjecture of Dowe [4] that no non-Bayesian method can be always invariant and statistically consistent while providing internally consistent parameter estimates. As a result of the reliance on Maximum likelihood, we suspect that simulation results would demonstrate over-fitting in the method of Hinton et. al.

6 Tests and IRAS LRS Spectral Classification Results

As a first step we have implemented mixture models with a single factor per class using total assignment, as described in section 5.1. The search process was based on that of Snob, with the addition of new heuristics for formation of initial models and sub-classes³.

6.1 Tests and Results on Simulated Data

We generated data based on the parameters used by Wallace and Freeman [20]. There were three trials of 1000 data sets of 200 observations per dataset, with five attributes per observation. Each data set consisted of 100 observations from a class with means μ_k all equal to 4 and 100 observations from a class with means μ_k all zero. All standard deviations σ_k were unity , and all factor load vectors \mathbf{a} were parallel to $(2, 3, 4, 5, 6)$. In all data in each of the three trials, the load vector lengths $|\mathbf{a}|$ were 1.5, 1.25 and 1.0 respectively, representing strong, weak, and barely detectable factors. These classes are moderately well separated, their variances in the fifth attribute being approximately 2 in the case of $|\mathbf{a}| = 1.5$. The means of various statistics and their standard errors are presented in table 1. In all cases the MML model had class structure very close to the true values; tabulated results for classes 1 and 2 refer to the estimated classes corresponding to true classes with means of 4 and 0 respectively. These results only apply to classes of datasets where MML preferred a factor model to the uncorrelated model.

With one exception, noted below, the results of the factor analysis concur with those of Wallace and Freeman [20], in terms of the mean squared length of $\hat{\mathbf{a}}$, $\hat{\beta}$ (where $\beta_k = a_k/\sigma_k$), and the factor load error vector $\hat{\mathbf{a}} - \mathbf{a}$, and the mean of the square of the sine of the angle between the true and estimated load vectors. As noted in Wallace and Freeman [20], these results outperform maximum likelihood techniques. For the barely detectable factor $|\mathbf{a}| = 1.0$, the factor load length is overestimated. This is presumed to be a selection effect, since for about one third of dataset classes an uncorrelated model was preferred, due to statistical variation obscuring the presence of a factor, with remaining statistical variation in the data included for analysis producing apparently stronger factors.

There is a discrepancy in the statistics of the $\sum_k \log \hat{\sigma}_k$, included to detect any bias in the estimation of the standard deviations (the true values of

³ See forthcoming Monash University School of Computer Science and Software Engineering technical report for details of these heuristics.

	$ \alpha = 1.5$		$ \alpha = 1.25$		$ \alpha = 1.0$	
	Class 1	Class 2	Class 1	Class 2	Class 1	Class 2
$\hat{\alpha}^2$	2.243 ± 0.016	2.226 0.016	1.583 0.013	1.580 0.014	1.129 0.010	1.120 0.010
$\hat{\beta}^2$	2.333 ± 0.020	2.310 0.019	1.674 0.016	1.745 0.086	1.214 0.013	1.197 0.012
$(\hat{\alpha} - \alpha)^2$	0.097 ± 0.002	0.098 0.002	0.102 0.002	0.109 0.007	0.102 0.003	0.111 0.006
\sin^2 error angle in $\hat{\alpha}$	0.031 ± 0.001	0.032 0.001	0.050 0.001	0.052 0.001	0.078 0.002	0.084 0.002
$\sum_k \log \hat{\sigma}_k$	-0.027 ± 0.006	-0.025 0.006	-0.033 0.006	-0.043 0.010	-0.070 0.007	-0.063 0.006
$\hat{\mu}_1$	3.9986 ± 0.0034	-0.0016 0.0033	3.9933 0.0053	-0.0011 0.0035	3.9975 0.0040	0.0020 0.0038
$\hat{\mu}_2$	4.0030 ± 0.0035	-0.0025 0.0036	3.9977 0.0054	-0.0002 0.0035	4.0000 0.0040	0.0012 0.0042
$\hat{\mu}_3$	3.9907 ± 0.0039	-0.0017 0.0039	3.9874 0.0055	-0.0029 0.0050	3.9865 0.0042	-0.0006 0.0042
$\hat{\mu}_4$	3.9851 ± 0.0041	-0.0004 0.0042	3.9818 0.0056	0.0007 0.0040	3.9830 0.0043	-0.0006 0.0043
$\hat{\mu}_5$	3.9968 ± 0.0045	-0.0019 0.0045	3.9936 0.0058	-0.0015 0.0045	3.9973 0.0046	0.0019 0.0048
Number of data sets estimated to have factors	998	998	964	957	669	690
Relative abundance (class 1)	0.4999 ± 0.0002		0.4994 0.0005		0.5000 0.0001	

Table 1. Estimates and errors on simulated data

which were all 1). Unlike MML estimation of single factors without mixture modelling, there appears to be a small but significant bias towards underestimating the standard deviations. Moreover, there may also be a very slight bias towards underestimating the means of class 1 (the class with ‘true’ means of 4). These effects are presumed to be a result of the inconsistency of total assignment (section 3.3), however it should be noted that the mean value of $\sum_k \log \hat{\sigma}_k$ for our estimator is considerably closer to zero than that obtained when using the maximum likelihood estimator out of the context of mixture modelling. This estimator shows means of approximately -0.113, -0.148 and -0.20 for $|\alpha| = 1.5, 1.25$ and 1.0 respectively [20].

6.2 Application: IRAS LRS Spectral Classification

We have also applied our program to astronomical point-source infrared spectra from the Infrared Astronomical Satellite (IRAS) Low Resolution Spectrometer (LRS). This data consists of spectra of 5425 point sources, in the wavelength range $7\text{ }\mu\text{m} - 23\text{ }\mu\text{m}$. Measurements were made with two instruments with overlapping wavelength coverage. The spectra as provided to the public by NASA have been corrected in a number of ways[1] to take into account calibration details and inconsistencies between the various components of the spectrometer, however many of the spectra still exhibit a lower reading from the red (long

wavelength) instrument than the blue (short wavelength) in the region of overlap despite measures to minimize this. For input to our modelling software, spectra were normalized to a mean attribute value of 1 to remove the very large variance in intensities from spectrum to spectrum.

A Bayesian maximum a posteriori classification of this data has been published by Goebel et. al [6]. This classification was produced by the AutoClass program [2] which assumes no correlation within classes. A total of 77 classes were found, with the distribution parameters of these classes being clustered into 'metaclasses'. As noted by Goebel et. al., there is considerable serial correlation in the attribute values (i.e. intensity values for given wavelengths).

In addition, there is large amount of correlation of the continuous kind which in our opinion is well modelled by factors. The AutoClass classification contains many classes which are identical to other classes apart from the presence of say a slightly larger broad peak at a certain wavelength. Rather than modelling the variance from spectral features of different strengths with large numbers of classes, and then trying to cluster them according to their distribution parameters, it is better to model them by a factor, where the factor scores correspond to the strength of the feature in a particular spectrum, and the factor loads indicate the effect the feature has on spectra.

Table 2 depicts the best classification yet found by our program for the IRAS LRS data. In each class, the points lie on the attribute means, the error bars extend one standard deviation above and below the means, and the factor loads are plotted as a line which usually crosses the horizontal axis around the middle. In addition to the parameters displayed as above, there are two lines representing $\mu_k + a_k$ and $\mu_k - a_k$, which can be identified as lines approximately following the shape of the mean points. These are included to give an indication of the range effects the factor can have for spectra with factor scores \pm one standard deviation from the mean of 0. There are a total of twelve classes.

The major difference between our classification and that of AutoClass is of course the large reduction in the number of classes. We believe this indicates that our classes are closer to the true class structure of the data. By and large, the factor in each class accounts for variation in the strength of some spectral feature : the $10\mu\text{m}$ and $10\mu\text{m}$ silicate emission bands in classes 2, 6 and 7, the $8\mu\text{m}$ band in classes 8, 9, and 10, and the colour temperature (which affects the overall slope of the curve) for the approximately blackbody spectra in classes 4, 5, 11 and 12. In many classes the factor accounts for the variation in the amount by which the intensities from the red and blue instruments miss in the overlap region : classes 4, 2 and 12 are strong examples of this. Also, in some classes the factor appears to account for variation in the *shape* of spectral features : in classes 6 and 7 in particular, the (negative) peak in the factor loads occurs at a lower wavelength than the peak in the means, and hence the factor score models the variation in peak emission wavelength as well as in the size of the peak.

For many classes the factor tries to model correlation structure which arises from more than one of the causes above. The strongest example of this is class 3, which would appear to simultaneously attempt to model features at $8\mu\text{m}$,

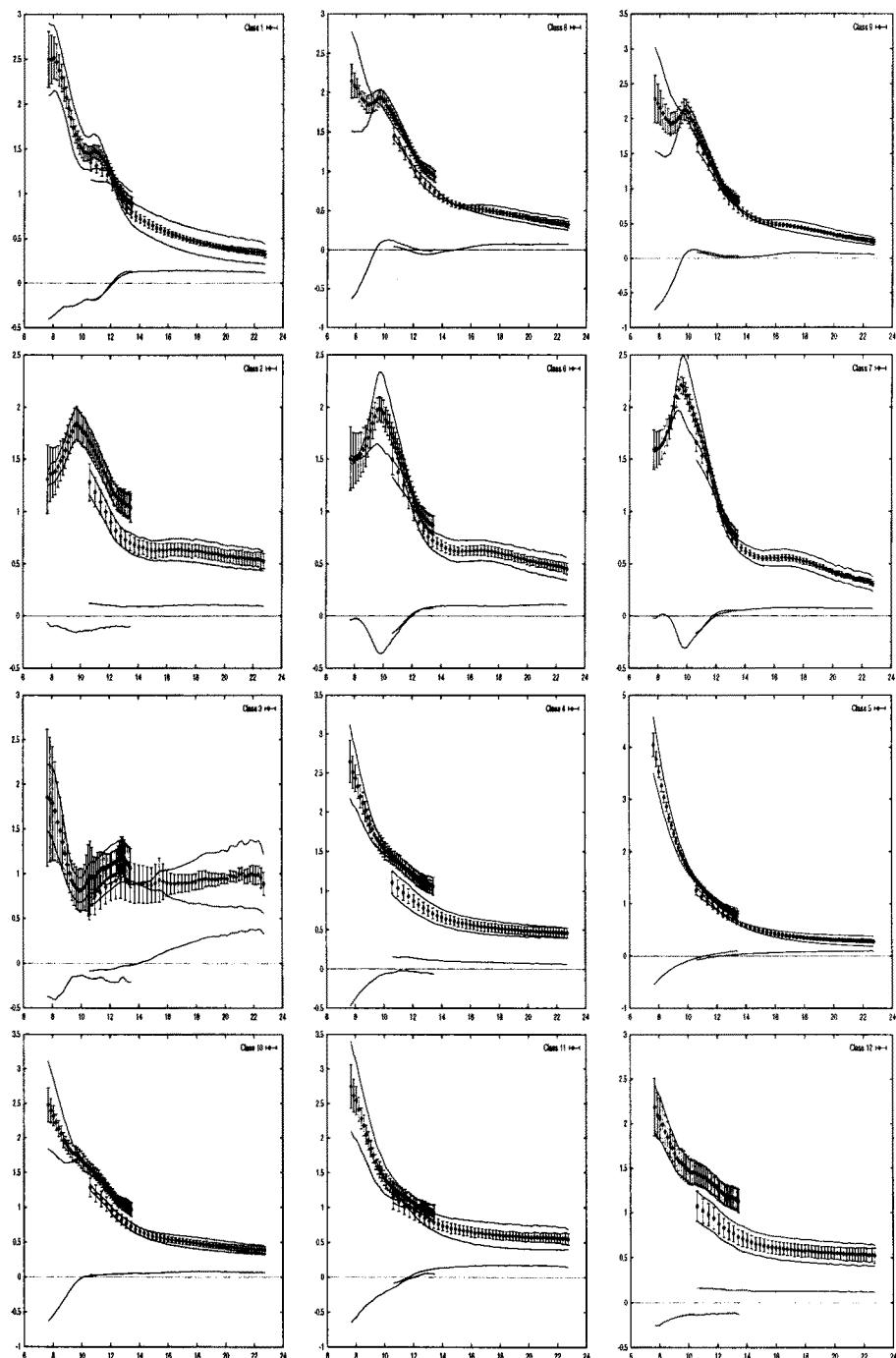


Table 2. IRAS LRS classes. The horizontal scale is in microns.

$10\mu\text{m}$, $11.5\mu\text{m}$, $13\mu\text{m}$, as well as the ‘plateau’ feature at $16\mu\text{m}$ - $22\mu\text{m}$, and the variation in the overlap region of the red and blue instruments. Judging by the large standard deviations, this class models a variety of sources which don’t fit well into other classes, and it is probable with a model incorporating multiple factors [15] per class that different factors and different classes would arise to account for the variety of effects currently modelled by a single factor.

It may be true that certain features are correlated with other features (meaning that a single factor will model both adequately), however this cannot be known without allowing multiple factor models. The most important difference made by allowing multiple factors would probably be the separation of the effects of red/blue instrument problem from the effects of true spectral features and colour temperature effects. In addition, a factor could be used to model overall intensity, removing the necessity for normalization and its inherent risk of loss of important information.

7 Future Work and Conclusion

We have presented a method for the inference of mixture models with a single factor per component. It has shown good results with test data, finding the correct class structure with good accuracy and concurring with Wallace and Freeman [20] for the factor estimates within classes. Our model represents a large improvement over plain mixture models for real world data, such as the IRAS LRS spectra, involving correlation other than that accounted for by separate sub-populations.

The methods presented for the incorporation of single factor models into mixture modelling with MML ought to extend to multiple factors, based on the work of Wallace [15]. More complex would be the ability to share factor estimates between multiple classes. These techniques would prove useful for the IRAS data.

In the mixture modelling of protein dihedral angles [3], many classes were found to lie on a line diagonal between the two attribute angles. It would be worthwhile to develop an MML estimator for factors involving angular data, based on the von Mises distribution [17], to model correlation in data such as this.

Acknowledgements

The second author was supported by Australian Research Council (ARC) Large Grants Nos. A49602504 and A49330656. We thank Chris Wallace for useful comments.

References

1. G. Beichman, H.J. Neugebauer, P.E. Clegg, and Y.J. Chester. IRAS catalogs and atlases : Explanatory supplement, 1988.

2. P. Cheeseman, M. Self, J. Kelly, W. Taylor, D. Freeman, and J. Stutz. Bayesian classification. In *Seventh National Conference on Artificial Intelligence*, pages 607–611, Saint Paul, Minnesota, 1988.
3. D.L. Dowe, L. Allison, T.I. Dix, L. Hunter, C.S. Wallace, and T. Edgoose. Circular clustering of protein dihedral angles by Minimum Message Length. In *Proc. 1st Pacific Symp. Biocomp.*, pages 242–255, HI, U.S.A., 1996.
4. D.L. Dowe, R.A. Baxter, J.J. Oliver, and C.S. Wallace. Point Estimation using the Kullback-Leibler Loss Function and MML. In *Proc. 2nd Pacific Asian Conference on Knowledge Discovery and Data Mining (PAKDD'98)*, Melbourne, Australia, April 1998. Springer Verlag. accepted, to appear.
5. D.L. Dowe and C.S. Wallace. Resolving the Neyman-Scott problem by Minimum Message Length. In *Proc. Computing Science and Statistics - 28th Symposium on the interface*, volume 28, pages 614–618, 1997.
6. J. Goebel, K. Volk, H. Walker, F. Gerbault, P. Cheeseman, M. Self, J. Stutz, and W. Taylor. A bayesian classification of the IRAS LRS Atlas. *Astron. Astrophys.*, (225):L5–L8, 1989.
7. H.H. Harman. *Modern Factor Analysis*. University of Chicago Press, USA, 2nd edition, 1967.
8. Hinton, Dayan, and Revow. Modelling the manifolds of images of handwritten digits. *IEEE Transactions on Neural Networks*, 8(1):65–74, 1997.
9. G.E. Hinton and R. Zemel. Autoencoders, minimum description length and Helmholtz free energy. In Cowan et. al, editor, *Advances in Neural Information Processing Systems*, 1994.
10. G.J. McLachlan and T. Krishnan. *The EM Algorithm and Extensions*. Wiley, New York, 1996.
11. J. Neyman and E.L. Scott. Consistent estimates based on partially consistent observations. *Econometrika*, 16:1–32, 1948.
12. F.M. Olnon and E. Raimond. IRAS catalogs and atlases : Atlas of low resolution spectra, 1986.
13. D.M. Titterington, A.F.M. Smith, and U.E. Makov. *Statistical Analysis of Finite Mixture Distributions*. John Wiley and Sons, Inc., 1985.
14. C.S. Wallace. Classification by Minimum Message Length inference. In G. Goos and J. Hartmanis, editors, *Advances in Computing and Information – ICCI '90*, pages 72–81. Springer-Verlag, Berlin, 1990.
15. C.S. Wallace. Multiple Factor Analysis by MML Estimation. Technical Report 95/218, Dept. of Computer Science, Monash University, Clayton, Victoria 3168, Australia, 1995. submitted to J. Multiv. Analysis.
16. C.S. Wallace and D.M. Boulton. An information measure for classification. *Computer Journal*, 11:185–194, 1968.
17. C.S. Wallace and D.L. Dowe. MML estimation of the von Mises concentration parameter. Technical report TR 93/193, Dept. of Computer Science, Monash University, Clayton, Victoria 3168, Australia, 1993. prov. accepted, Aust. J. Stat.
18. C.S. Wallace and D.L. Dowe. MML mixture modelling of Multi-state, Poisson, von Mises circular and Gaussian distributions. In *Proc. 6th Int. Workshop on Artif. Intelligence and Stat.*, pages 529–536, 1997. An abstract is in Proc. Sydney Int. Stat. Congr. (SISC-96, 1996), p197; and also in IMS Bulletin (1996), 25 (4), p410.
19. C.S. Wallace and P.R. Freeman. Estimation and inference by compact coding. *Journal of the Royal Statistical Society (Series B)*, 49:240–252, 1987.
20. C.S. Wallace and P.R. Freeman. Single factor analysis by MML estimation. *Journal of the Royal Statistical Society (Series B)*, 54:195–209, 1992.

Discovering Associations in Spatial Data - An Efficient Medoid Based Approach*

Vladimir Estivill-Castro¹ and Alan T. Murray²

¹ Department of Computer Science and Software Engineering, Faculty of Engineering, The University of Newcastle, Callaghan NSW 2308, Australia

² Australian Housing and Urban Research Institute, Department of Geographical Sciences and Planning, The University of Queensland, Brisbane QLD 4072, Australia

Abstract. Spatial data mining is the discovery of novel and interesting relationships and characteristics that may exist implicitly in spatial databases. The identification of clusters coupled with Geographical Information System provides a means of information generalization. A variety of clustering approaches exists. A non-hierarchical method in data mining applications is the medoid approach. Many heuristics have been developed for this approach. This paper carefully analyses the complexity of hill-climbing heuristics for medoid based spatial clustering. Improvements to recently suggested heuristics like CLARANS are identified. We propose a novel idea, the stopping early of the heuristic search, and demonstrate that this provides large savings in computational time while the quality of the partition remains unaffected.

1 Introduction

The emergence of Geographical Information Systems (GIS) with convenient and affordable methods for storing large amounts of spatial data has accelerated the rate at which spatially referenced information is collected in electronic and magnetic media. Aerial photography and satellite remote sensing have contributed to an explosion of geographic data production. Geographic data sets are now being generated faster than they can be analyzed. Currently, GIS is designed and used for confirmatory rather than exploratory analysis. What has been found to be very lacking in GIS is analytical tools for exploring spatial relationships.

Spatial data mining [7, 12, 2] is the discovery of novel and interesting relationships and characteristics that may exist implicitly in spatial databases. Clustering detection in spatially referenced data provides a means of generalization that is complementary to techniques for generalization used in data mining in relational databases [1]. Clustering is the task of identifying groups in a data set based upon some criteria of similarity. Moreover, for geo-referenced information there is an explicit measure of similarity that is relatively well defined. Clustering, or cluster analysis, has a direct interpretation for knowledge extraction in this context [3, 5, 12, 15].

* This research was supported in part by a grant from the Australian Research Council.

The following example illustrates some of the roles for clustering in spatial data mining. Let us assume that amongst many layers of data, Fig. 1 (a) is a layer of information that stores the location of three types of crimes (stolen vehicles, break-ins, and robberies). Other information layers for this region are

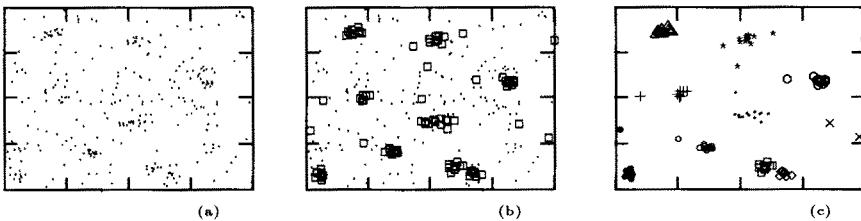


Fig. 1. (a) crimes occurrences, (b) highlighted location of stolen vehicles and (c) clustering via LOCAL HILL CLIMBING.

the location of churches (Fig. 2 (a)), the location of parks (Fig. 2 (b)), and the location of the subway stations (Fig. 2 (c)).

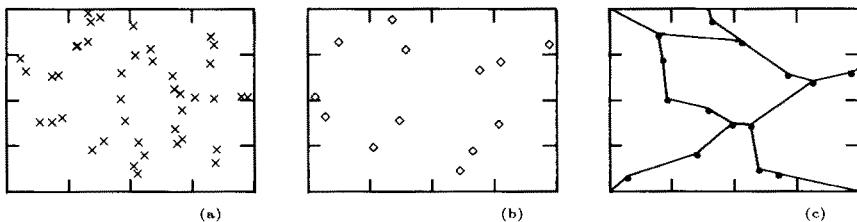


Fig. 2. (a) Location of churches, (b) centroids of parks, and (c) subway lines with subway stations.

A GIS analysis tool for exploring these and other potential layers for this region is a mining agent that will autonomously search for a localized pattern or data anomaly. The exploration engine may seek a rule or some link or association with the occurrence of stolen cars. Thus, a selection on attribute value $\text{crime_type}=\text{stolen_car}$ corresponds to highlighting the locations of stolen vehicles as shown in Fig. 1 (b). These selected points can be clustered by the techniques described here or the slower algorithms presented elsewhere [9, 12, 15]. In particular, a medoid-based approach results in the clustering illustrated in Fig. 1 (c). The medoids of this clustering are illustrated in Fig. 3 (a). The mining agent then will overlay the medoids with each of the other 3 thematic data layers of Fig. 2. By computing the average of the distances from medoids to closest point in each layer, the agent finds that the distance is unusually small when the medoids and the train stations are overlaid (refer to Fig. 3(a)-(c)). This allows the miner to discover a relationship between the location of stolen cars and subway stations.

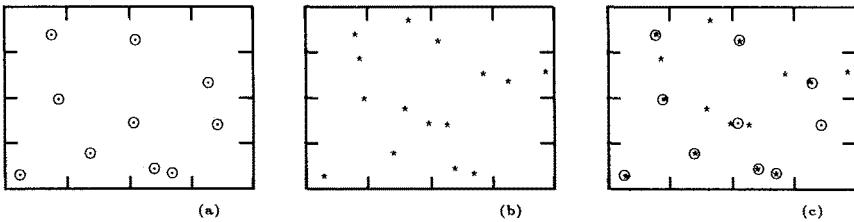


Fig. 3. (a) Medoids with LOCAL HILL CLIMBING, (b) subway stations, and (c) overlay.

In the process of finding a potential association or anomaly, the mining agent may cluster many data layers or selected subsets (on one or several common attribute values as in the *crime_type=stolen_car* presented before). Thus, it is crucial that clustering be performed efficiently. In this paper we carefully examine the complexity of hill-climbers for medoid based clustering. As a result we improve upon the approach used in spatial data mining algorithms like CLARANS [12] and others [9, 15]. Moreover, we review the uses of clustering for spatial data mining and formulate new efficient algorithms for discovering associations between spatially referenced data.

Section 2 describes the formulation of the medoid based approach. Section 3 describes the algorithms and discusses their complexity. Section 4 presents experimental evaluation of the complexity of the algorithms and the insight into the strategy proposed in Sect. 5. Further refinements are presented in Sect. 6. We conclude with final remarks in Sect. 7.

2 Medoid Based Clustering

A number of clustering methods have emerged from the fields of statistics, geography, machine learning, knowledge discovery and operations research [11]. In the field of Knowledge Discovery, density based approaches have been proposed [3] as an alternative to non-hierarchical clustering [12, 15] on the grounds of memory efficiency. It has recently been shown that this argument is misleading [4, 11]. Moreover, density based approaches [3] require sophisticated determination of density parameters with somewhat limited automation.

Formulations of non-hierarchical clustering problems vary by the criterion that measures partition quality and usually correspond to some evaluation of the within group difference versus the distinctness among clusters. However, certain approaches do have advantages over others, and in particular, the medoid approach offers robustness with respect to noise as well as a structure which can be solved approximately by several techniques [4, 11]. We now describe the medoid approach.

Consider a set of data items $S = \{s_1, s_2, \dots, s_n\}$ where each s_i is a point in d -dimensional real space \mathbb{R}^d . The clustering problem consists of naturally grouping these points into k clusters. A common approach to clustering is to identify a

representative for each cluster and to assess the quality of the clustering as the average distance between items and their representative. In the medoid approach, the set C of representatives is restricted to be a subset of S . Thus, the clustering problem translates to a combinatorial optimization problem where the goal is a set of representatives that minimizes the following criterion MP defined by

$$\text{MP}(C) = \sum_{i=1}^n d(s_i, \text{rep}[s_i, C]) \quad (1)$$

over all subsets C of S with $\|C\| = k$. Usually, the distance d corresponds to a Minkowski distance (i.e. $d_p(\mathbf{x}, \mathbf{y}) = (\sum_{u=1}^d |x_u - y_u|^p)^{1/p}$). Let $\text{rep}[s_i, C]$ denote the closest point in $C = \{m_1, \dots, m_k\}$ to s_i , and thus, $\min_{j \in \{1, \dots, k\}} d(s_i, m_j) = d(s_i, \text{rep}[s_i, C])$. The value $\text{MP}(C)$ represents the quality of the clustering C .

3 Solution Heuristics

Obtaining an optimal solution (a set of medoids that globally minimizes MP) is unrealistic for large data sets. The medoids approach is known to be NP-complete. Thus, heuristic approximations are essential and have received considerable attention in the literature [9, 10]. For spatial information it is assumed that n is large, the dimension d is 2 or 3 and the number k of clusters is small (10-50). Thus, d and k can be considered constant. We use $f(n) = O(g(n))$ to denote that $f(n)$ is bounded from above by $c g(n)$ for some constant c . We use $f(n) = \Omega(g(n))$ to denote that $f(n)$ is bounded from below by $c g(n)$ for some constant c . We use $f(n) = \Theta(g(n))$ when $f(n) = \Omega(g(n))$ and $f(n) = O(g(n))$.

The search space X for MP is the set of all subsets $C \subseteq S$ where $\|C\| = k$ (thus, $X \subset 2^S$). The objective function $\text{MP}(C) : X \rightarrow \mathbb{R}$ assigns a quality value to each subset $C \subseteq S$ as given by (1). Recall that the elements of C are taken as representatives and each s_i belongs to its nearest representative. Heuristically searching X for the smallest value $\text{MP}(C)$ can be organized by providing X with the structure of a graph whose set of nodes is the set X . Two nodes C and C' are adjacent if they differ in exactly one medoid (that is, $\|C \cap C'\| = k-1$). Local Search [8] in this graph is computationally feasible because computing $\text{MP}(C')$ on an adjacent node C' of C requires $\Theta(n)$ time ($\Theta(n)$ time to find $\text{rep}[s_i, C']$ for $i \in \{1, \dots, n\}$, and $\Theta(n)$ time to compute $\text{MP}(C')$ as defined in (1)). Medoids offer another advantage. The value $\text{MP}(C')$ need not be computed, rather the gradient $\Delta(C, C') = \text{MP}(C) - \text{MP}(C')$ between C and an adjacent node C' can be used because it can be computed in $\Theta(n)$ time with a smaller constant.

The most reputed heuristics for MP are essentially rediscoveries of each other because they are instances of *Local Search*. Such *Local-Search* heuristics begin with a randomly selected solution C_0 . Iteratively, the heuristic explores a subset W of adjacent nodes of a current node C_t and moves to an adjacent node C_{t+1} . The move from C_t to C_{t+1} consists of interchanging one medoid $s_m \in C_t$ for one observation $s_u \in S - C_t$, so $C_{t+1} = C_t \cup \{s_u\} - \{s_m\}$. Typically, $\text{MP}(C_t) > \text{MP}(C_{t+1})$ and the search halts when such a move is no longer possible. A local

optima is reached when no interchange of a medoid for an observation results in an improved solution. The characteristics of various Local-Search heuristics to optimize MP are summarized in Table 1.

Table 1. Characteristics of Local-Search heuristics.

LOCAL SEARCH	Strategy to move from C_t to $C_{t+1} = C_t \cup \{s_u\} - \{s_m\}$	Time complexity to improve MP value	Guarantees local optima
GLOBAL HILL CLIMBING [6, 9, 12, 13]	Move to the best of the $\Theta(n)$ neighbors of C_t	$\Theta(n^2)$	yes
RANDOMIZED HILL CLIMBING [9, 12]	Move to the best of m sampled neighbors of C_t	$\Theta(\beta n)$	no
LOCAL HILL CLIMBING [10, 13]	Move to a new neighbor as soon as it is found.	$\Theta(\delta_t n)$	yes
DISTANCE RESTRICTED HILL CLIMBING [9, 14, 15].	Move to the best neighbor among those with $d(s_u, s_m) < \rho$	$\Theta(\rho_t n)$	no

Table 1 reveals that LOCAL HILL CLIMBING is the best local search heuristic. Knowledge Discovery has not recognized this fact. Convergence to a local optima for GLOBAL HILL CLIMBING (and thus PAM [9]) is $O(N_G n^2)$ time where N_G is the total number of improving cycles. RANDOMIZED HILL CLIMBING (and thus CLARANS [12]) requires $O(N_R \beta n)$ time where N_R is also the number of improving cycles. The total time for the convergence to a local optima for LOCAL HILL CLIMBING is $O(N_L n)$ where N_L are its number of improvement cycles.

The heuristics in Table 1 were coded in C. Most of the code is common, but the loop that moves form C_t to C_{t+1} is not. Test sets have been constructed as follows. Select τ points $\mathbf{c}_1, \dots, \mathbf{c}_\tau$ randomly and uniformly in $[0, 1] \times [0, 1]$. These points are virtual centers that are not included in the data set. The smallest separation $D = \min_{i \neq j} d(\mathbf{c}_i, \mathbf{c}_j)$ is used to determine a common virtual radius $r = D/2$. With this information, as many as $(1 - N)n$ data points are chosen by first selecting randomly a virtual centroid \mathbf{c}_i and then selecting polar coordinates γ_i, ψ_i (γ_i is uniform in $[0, r]$ and ψ_i is uniform in $[0, 2\pi]$). The point in the data set is $\mathbf{c}_i + (\gamma_i \sin \psi_i, \gamma_i \cos \psi_i)$. The data set is shuffled with Nn randomly and uniformly selected points where $N \in [0, 1]$ is a percentage of the amount of noise.

For these data sets, Table 2 presents the average CPU time in seconds of 10 executions (and the corresponding 95% confidence interval) for three heuristic methods. LOCAL HILL CLIMBING proves to be far more efficient than approaches presented recently in the Data Mining literature.

4 Complexity Analysis

LOCAL HILL CLIMBING incorporates an adaptive search strategy to explore only a fraction of the adjacent nodes of the current solution C_t . In most cases, this reduces to $O(n)$ the time to decrease the value of MP. This is an improvement of a factor of n over the time complexity of GLOBAL HILL CLIMBING. To

Table 2. Solution time in CPU seconds.

n	LOCAL HILL CLIMBING		RANDOMIZED HILL CLIMBING (used in CLARANS [12])		GLOBAL HILL CLIMBING (used in PAM [9])	
100	10.1	\pm 2.0	44.2	\pm 4.2	61	\pm 8.1
200	41.2	\pm 5.3	204.3	\pm 32.4	350.3	\pm 60.9
500	338.2	\pm 10.3	1435.3	\pm 123.8	2318.7	\pm 240.2
1000	939.2	\pm 42.3	-	-	-	-

achieve this, the observations are enumerated as s_i , for $i = 1, \dots, n$. Each s_i that is not a medoid, is evaluated for a possible exchange with a medoid. This is $O(kn) = O(n)$ time per observation s_i when k is small and n is large. After an improvement, the exploration is restarted with s_{i+1} (with s_1 following s_n). A full loop around the observations with no improvement gives the local optima and the search halts.

Note that the time complexity of the step C_t to C_{t+1} for LOCAL HILL CLIMBING is adaptive to the shape of the objective function. It requires $O(\delta_t n)$ time, where $s_{(i+\delta_t) \bmod n}$ is the next observation where an improvement occurs after the improvement that occurred at s_i . Note also that the neighbors explored to improve the solution C_t is a set W of dynamic size δ_t . In the best case, δ_t is as small as 1, while at the local optima it consists of all adjacent nodes and has size $\Omega(n)$.

Because LOCAL HILL CLIMBING exploits the sharpness of the objective function, it is much more efficient than GLOBAL HILL CLIMBING. When a medoid is on an outlier, such solution has many adjacent nodes that are a significant improvement to its MP value. So, at least for outliers, it is easy to find improvements and LOCAL HILL CLIMBING profits from this.

Although LOCAL HILL CLIMBING represents an improvement of $\Theta(n)$ magnitude in time requirements, a more detailed analysis of its behavior suggests that more efficiencies may be identified. Table 3 shows the results of experiments to estimate $N_L/n = \sum_{t=1}^k \delta_t/n$. Ten different data sets varying in size ($n = 50, 100, 200, 500, 1000$ and 5000) were generated. Each was clustered 10 times (new C_0 each time), into k groups for $k = 2, 4, 6, 8, 10, 12$ and 14 . The experiments show that N_L rarely exceeds $4n$. N_L appears independent of k , and the dominant term in the total complexity of LOCAL HILL CLIMBING seems to be $4n^2$, also independently of k . Moreover, $\sum_{t=1}^k \delta_t/n$ is independent of n and $k!$ Thus, the number of times a data point is analyzed for an interchange is constant. This is clearly superior to the other hill-climbing approaches, since it implies that equivalent solutions are found with a neighborhood of amortized constant size. In contrast, CLARANS [12] uses a neighborhood of size the largest of 250 and $k(n - k)$. Further, the total time of an execution of LOCAL HILL CLIMBING hardly is enough for GLOBAL HILL CLIMBING to take 4 improving steps (from C_0 to C_4) towards a potentially local optima still $\Omega(N_G)$ steps away.

Given these findings, it is no surprise that previous research for MP optimization has focused on LOCAL HILL CLIMBING. However, for Knowledge Discovery,

Table 3. Experimental bounds for N_L/n .

n	Λ	% of times Λ cycles are needed						
		2	4	6	8	10	12	14
100	$\Lambda = 2$	60%	61%	32%	48%	24%	30%	10%
	$\Lambda = 3$	40%	39%	50%	44%	64%	51%	74%
	$\Lambda = 4$			12%	8%	12%	11%	11%
	$\Lambda = 5$			5%				5%
200	$\Lambda = 2$	80%	60%	55%	36%	22%	14%	42%
	$\Lambda = 3$	20%	40%	45%	60%	64%	56%	40%
	$\Lambda = 4$				4%	14%	24%	18%
	$\Lambda = 5$						6%	
500	$\Lambda = 1$							14%
	$\Lambda = 2$	44%	64%	34%	18%	34%	22%	6%
	$\Lambda = 3$	40%	36%	61%	70%	50%	68%	52%
	$\Lambda = 4$	16%		5%	12%	16%	10%	28%
1000	$\Lambda = 1$			14%				
	$\Lambda = 2$	64%	51%	52%	14%	20%	16%	18%
	$\Lambda = 3$	30%	49%	28%	64%	64%	78%	62%
	$\Lambda = 4$	2%		4%	16%	10%	6%	14%
	$\Lambda = 5$	3%		2%		5%		6%
	$\Lambda = 6$	1%				1%		
5000	$\Lambda = 2$	9%	31%	30%	20%	34%	12%	58%
	$\Lambda = 3$	91%	66%	64%	65%	53%	77%	35%
	$\Lambda = 4$		3%	6%	15%	10%	15%	7%
	$\Lambda = 5$					3%		

MP optimization may be less important than to improve the efficiencies of LOCAL HILL CLIMBING. The optimal value of $MP(C)$ is not required, rather what is required is that the partition defined using C representatives results in a Voronoi Diagram whose components closely reflect the spatial patterns in the data. It is not hard to imagine two different sets of medoid solutions such that MP is much worse in one case than the other but for which the partitions are equivalent. For spatial pattern spotting one may be willing to sacrifice quality in the final values of $MP(C)$ if in fact the partition is close enough to make the pattern known.

The first conclusion to be drawn is that $N_L \approx 4n$. The value of the last δ_t is $\Omega(n)$ because of the guarantee of a local optimum. Thus, in the last round, the search is not modifying clusters, rather certifying that the current partition is a local optimum. Avoiding this last round would result in 25% less computational effort (N_L would be $3n$).

It is potentially feasible, then, to monitor three quantities to detect that improvements in MP are no longer improvements in the clustering.

1. When moving from C_t to $C_{t+1} = C_t \cup \{s_u\} - \{s_m\}$, the value $dist_t = d(s_u, s_m)$ is potentially small with respect to earlier values.
2. The value of $\Delta_{t+1} = \Delta(C_t, C_{t+1})$ is potentially small with respect to earlier values (like $\Delta_1 = \Delta(C_0, C_1)$).
3. The value of δ_t is potentially large, say $\Omega(n)$ while $\delta_1 = \Theta(1)$.

We have conducted experiments to reveal the behavior of these parameters.

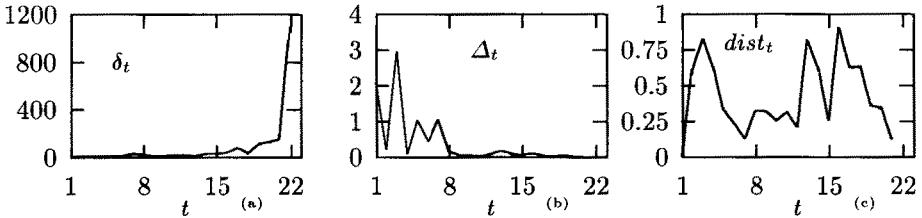


Fig. 4. The performance parameters (a) δ_t (b) Δ_t and (c) $dist_t$ in one execution of LOCAL HILL CLIMBING.

Figure 4 shows their behavior in one execution with the highlighted points of Fig. 1 (b) that produces the clustering of Fig. 1 (c). Although the behavior for neither of them is monotone on a single execution, it is clear that the early values of δ_t are very small and they sharply grow by at least an order of magnitude when convergence has been reached (refer to Fig. 4 (a)). In fact, more computation time is spent on certifying local optimality of MP at $t = 22$ (5.5 CPU seconds) than the total time to find the partition (4.45 CPU seconds). This clearly illustrates that in this particular execution, stopping at $t = 21$ would have resulted in the same clustering and it would have used less than 50% of the time. Figure 4 (b) shows also that Δ_t has a tendency to be monotonically decreasing. In fact, in this example, $\Delta_{21} = 0.31 \times 10^{-2}$ and $\Delta_{22} = 0.18 \times 10^{-2}$ (and obviously at convergence $\Delta_{23} = 0$, while for three values of t with $t \leq 9$ we have Δ_t is at least 100 times larger).

The behavior of $dist_t$ is oscillatory and does not have the tendency as the other two. This indicates that improvements in the clustering are not to be expected from a distance restricted neighborhood and explains why DISTANCE RESTRICTED HILL CLIMBING has been found to be less effective than other methods [14].

In order to confirm that the behavior of δ_t and Δ_t have the tendencies just described across different settings we have repeated the experiments for each $n = 50, 100, 200, 500, 1000$ and 5000 ,

- 10 times but different initial set C_0 ,
- 10 different data sets
- finding 2, 4, 6, ... 14 clusters.

The expected behavior for $k = 10$ with $n = 1000$ is typical of the expected behavior of other values of k and n and is shown in Fig. 5. The plot for δ_t is shown from the last t downwards. Thus, the larger values in the x are earlier times and the first position in the plot ($x = 1$) is the average size of the last δ_t in all executions with $k = 10$. It should be evident from this plot that for the first half of the interchanges δ_t is small while Δ_t represents a sizable improvement in MP. However, in the late half of the interchanges, Δ_t is a very small improvement on MP and δ_t can become very large, in particular, the last δ_t is a costly effort to guarantee local optimality.

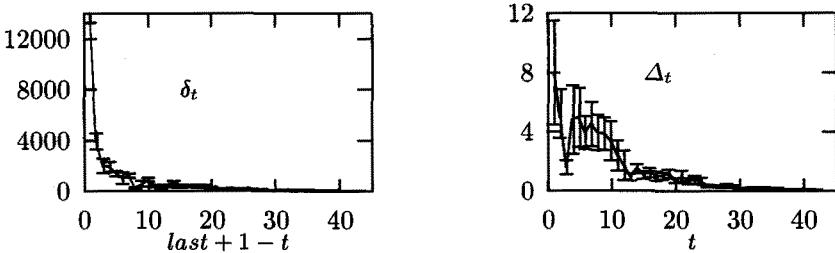


Fig. 5. Expectation of performance parameters δ_t and Δ_t with 95% confidence intervals ($n = 1000$).

5 Premature Termination

The detailed analysis presented before allows us to propose a new version of LOCAL HILL CLIMBING for spatial clustering that is more efficient because it stops much earlier than the standard version but the quality of the clustering is perturbed only slightly.

Our strategy is to stop when

- there is a sharp increase from δ_t to δ_{t+1} after 3 rounds over the data set (early increases in δ_t are normal), or
- there is a sharp increase of an order of magnitude in δ_t or $\delta_t > n/k$ regardless of being early in the search.

To demonstrate the effectiveness of this strategy, we have once more performed simulations. This time, we compare

- the clustering of LOCAL HILL CLIMBING with our stopping strategy and
- with the clustering obtained if the same execution was allowed to run until a local optima for MP is found.

We do not compare values of MP to compare the quality of the clustering obtained by stopping early, since the difference is of the magnitude of late Δ_t . We compare the more relevant aspect of classification accuracy of the earlier clustering with respect to the final clustering. This is the real difference in terms of pattern spotting and grouping. We take the final clustering as a tutor and every data point placed in another group by the earlier clustering is a classification error. The classification accuracy Acc is then given as follows.

$$Acc(C) = \frac{\# \text{ of data points} - \# \text{ of errors}}{\# \text{ of data points}} = \frac{n - \# \text{ of errors}}{n}.$$

We carried out experiments for each value of $k = 6, 8, \dots, 14$. The LOCAL HILL CLIMBING heuristic was run 10 times on 10 data sets with sizes $n = 100$, $n = 500$ and $n = 1000$ and 10 clouds with 10% noise. Because of space limitation we present the values of $Acc(C)$ for $k = 8, 10$ and 12 , since they are typical in Table 4. Table 4 also reports the time used with our stopping strategy as a percentage of the time used by LOCAL HILL CLIMBING. The 95% confidence intervals are also shown to illustrate that our results hold across the different data sets and different starting sets C_0 .

Table 4. Accuracy and CPU time requirements of our premature termination.

n	$k = 8$		$k = 10$		$k = 12$	
100	Accuracy 99.7 ± 0.2	CPU time 22% ± 1.8	Accuracy 98.9 ± 0.5	CPU time 31% ± 3.2	Accuracy 99.2 ± 0.4	CPU time 32% ± 3.7
500	Accuracy 99.2 ± 0.1	CPU time 30% ± 3.7	Accuracy 99.6 ± 0.1	CPU time 39% ± 2.0	Accuracy 98.8 ± 0.4	CPU time 33% ± 5.2
1000	Accuracy 98.2 ± 0.5	CPU time 27% ± 5.0	Accuracy 99.2 ± 0.1	CPU time 35% ± 6.0	Accuracy 99.0 ± 0.1	CPU time 34% ± 2.2

6 Two Further Refinements

In this section we present two further refinements derived from the understanding of the complexity of LOCAL HILL CLIMBING. The first refinement deals with the format of the data. In some cases, we may have sites or data already somewhat generalized by its representation in a tessellation. For example, the information of the exact point in space of stolen cars has been generalized to the district level. However, it is possible to perform non-hierarchical clustering almost as before. Conceptually, if T_α is the integer value for district α , then we can consider that we have T_α instances located in the center of mass of district α . How is this impacted by the generalization into districts? The medoid criterion as defined in (1) is extended to

$$\text{MP}_{\text{weight}}(C) = \sum_{\alpha=1}^n T_\alpha d(s_\alpha, \text{rep}[s_\alpha, C]), \quad (2)$$

where s_α is the centroid of district α and we use n for the number of districts. Then, the subroutine to compute $\Delta(C, C')$ does not have a significant change since

$$\begin{aligned} \Delta_{\text{weight}}(C, C') &= \text{MP}_{\text{weight}}(C) - \text{MP}_{\text{weight}}(C') \\ &= \sum_{\alpha=1}^n T_\alpha [d(s_\alpha, \text{rep}[s_\alpha, C]) - d(s_\alpha, \text{rep}[s_\alpha, C'])]. \end{aligned}$$

The computation of $\Delta(C, C') = \text{MP}(C) - \text{MP}(C')$ requires $\Theta(n)$ time where n is only the number of districts (and not the number of sites) and the constant factor under the Θ is small. In fact, the coefficients T_α can be real numbers incorporating other attribute information about the site (for example, the commercial value of the stolen cars). Thus, we see that the approach discussed here efficiently incorporates other aspects of the geo-referenced data without penalty to its computational resources.

Our second refinement deals with the fact that the clustering is used in exploring overlays of data. Consider the clustering of stolen cars in Fig. 1. What if the location of churches or the location of subway stations had been used to select representatives for clusters of stolen cars and to initialize the clustering of stolen

cars with these representatives? We now show that this results in computational savings because one layer has much less data points.

More formally, the clustering problem is extended in the following way. Consider two sets S and R of data items; that is $S = \{s_1, \dots, s_n\}$ and $R = \{r_1, \dots, r_m\}$ are sets of points in d -dimensional space \mathbb{R}^d . The medoid-based clustering of S into k groups *with respect to* R is to find the set $C \subseteq R$ with $\|C\| = k$ that minimizes $\text{MP}(C) = \sum_{i=1}^n d(s_i, \text{rep}[s_i, C])$. In the discussion up to here we considered $S = R$. However, this other formulation shows that this not need to be the case. As just indicated, S could be the location of stolen cars while R could be the location of churches or the location of subway stations.

The algorithms and techniques discussed apply directly to this extended formulation or in combination with the first refinement of this section. To compute $\Delta(C, C')$ remains $\Theta(n)$ as just illustrated in this section. However, $N_L = \sum \delta_t$ is no longer $4\|S\| = 4n$ but $4\|R\| = 4m$. This is a significant improvement when $n \ll m$. In our running example, the number of locations for stolen cars is 101 but the number of subway stations is 15. An initial clustering of the stolen cars into 10 groups can be computed by clustering these locations with respect to the subway stations in time that is an order of magnitude less than if the stolen cars locations are considered in isolation. This initial clustering can be taken as the C_0 clustering for the stolen car layer and we apply LOCAL HILL CLIMBING with our stopping strategy. When there is a relationship, this initial clustering is quickly identified as sufficient by the stopping strategy and the association between stolen cars and subway stations is found much faster.

In general, once a layer L_1 has been clustered with representative set M_1 , it is efficient to save M_1 (since space wise is much smaller than L_1). Then, performing a clustering of another layer L_2 with respect to M_1 allows fast identification or elimination of the potential association between L_1 and L_2 . Again, our algorithm easily extends to handle this refinement. In fact, monitoring δ_t and Δ_t after the initialization of C_0 from another layer is the device to early identify or discard associations across two layers. Thus, our approach has additional benefits in the overall exploration of several layers.

7 Final Remarks

We have shown that LOCAL HILL CLIMBING (a well-known heuristic for approximately solving the p -median problem) is a sound and efficient approach for clustering spatial data because of its natural interpretation, its robustness to noise and its computational complexity. Moreover, we have shown than in the environment of Knowledge Discovery applications where the number n of sites is large (certainly above 100), the goal is to obtain a natural clustering rather than the best value of MP .

By carefully analyzing the complexity of LOCAL HILL CLIMBING and its adaptive resizing of a window of neighbors we have discovered that LOCAL HILL CLIMBING can be stopped early without impact on the quality of the clustering, but with savings that reduce the computational effort to a third. The experiments

presented show at least 97% accuracy in clustering 1000 locations with 10% noise (100 elements do not really have a preferable cluster). This figures are impressive. We have shown that the magnitude Δ_t of the MP increments is relatively small for the last half of them while the time δ_t to take an improvement is small for the first half. The δ_t is costly at the end. This clearly explains why the experiments show the impressive accuracy, namely, most of the work of finding a good partition is done in the first half of the improving steps. Because during these improvements δ_t is small, the relevant work is done in the first third of the CPU time. We avoid the overhead.

References

1. Y. Cai, N. Cercone, and J. Han. Attribute-oriented induction in relational databases. In G. Piatetsky-Shapiro and W.J. Frawley, *Knowledge Discovery in Databases*, 213–228, Menlo Park, CA, USA, 1991. AAAI Press.
2. M. Ester, H.P. Kriegel, and J. Sander. Spatial data mining: A database approach. In A. School and A. Voisrad, *Advances in Spatial Databases, 5th Int. Symp., SDD-97*, 47–66, Berlin, Germany, 1997. Springer-Verlag LNCS 1262.
3. M. Ester, H.P. Kriegel, S. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In E. Simoudis, J. Han, and U. Fayyad, *Proc. of the 2nd Int. Conf. on Knowledge Discovery and Data Mining (KDD-96)*, 226–231, Menlo Park, CA, 1996. AAAI, AAAI Press.
4. V. Estivill-Castro and A.T. Murray. Mining spatial data via clustering. Technical Report #5/97, FIT, QUT, Brisbane 4000, Australia, 1997.
5. A. Ghozeil and D.B. Fogel. Discovering patterns in spatial data using evolutionary programming. In J.R. Koza, *Genetic Programming: Proc. of the First Annual Conf.*, 521–527, Cambridge, MA, 1996. MIT Press.
6. M Goodchild and V. Noronha. Location-allocation for small computers. Monograph 8, University of Iowa, 1983.
7. K. Han, J. Koperski and N. Stefanovic. GeoMiner: A system prototype for spatial data mining. *SIGMOD Record*, 26(2):553–556, 1997.
8. D.S. Johnson, C.H. Papadimitriou, and M. Yanakakis. How easy is local search? *Journal of Computer System Sciences*, 37:79–100, 1988.
9. L. Kaufman and P.J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons, NY, US, 1990.
10. A.T. Murray and R.L. Church. Applying simulated annealing to location-planning models. *Journal of Heuristics*, 2:31–53, 1996.
11. A.T. Murray and V. Estivill-Castro. Cluster discovery techniques for exploratory spatial data analysis. *Int. J. of GIS* (to appear).
12. R.T. Ng and J. Han. Efficient and effective clustering methods for spatial data mining. In J. Bocca, M. Jarke, and C. Zaniolo, *Proc. of the 20th Conf. on Very Large Data Bases (VLDB)*, 144–155, San Francisco, CA, 1994. Morgan Kaufmann.
13. D. Rolland, E. Schilling and J. Current. An efficient tabu search procedure for the p -median problem. *European Journal of Operations Research*, 96:329–342, 1996.
14. P. Sorensen. Analysis and design of heuristics for the p -median location-allocation problem. Master's thesis, Dep. of Geography, U. California, Santa Barbara, 1994.
15. T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH:an efficient data clustering method for very large databases. *SIGMOD Record*, 25(2):103–114, June 1996.

Data Mining Using Dynamically Constructed Recurrent Fuzzy Neural Networks

Yakov Frayman and Lipo Wang

Deakin University, School of Computing and Mathematics,
662 Blackburn Road, Clayton, Victoria 3168, Australia
E-mail: yfraym@deakin.edu.au, lwang@deakin.edu.au

Abstract. Approaches to data mining proposed so far are mainly symbolic decision trees and numerical feedforward neural networks methods. While decision trees give, in many cases, lower accuracy compared to feedforward neural networks, the latter show black-box behaviour, long training times, and difficulty to incorporate available knowledge. We propose to use an incrementally-generated recurrent fuzzy neural network which has the following advantages over feedforward neural network approach: ability to incorporate existing domain knowledge as well as to establish relationships from scratch, and shorter training time. The recurrent structure of the proposed method is able to account for temporal data changes in contrast to both feedforward neural network and decision tree approaches. It can be viewed as a gray box which incorporates best features of both symbolic and numerical methods. The effectiveness of the proposed approach is demonstrated by experimental results on a set of standard data mining problems.

1 Introduction

Data mining is one of the most promising fields for application of intelligent information processing technologies. A main aim of data mining is to extract useful patterns and nontrivial relationships from large collections of data records [3], [5], [10], [17]. This method can provide users with a powerful tool for exploiting vast amount of stored data.

The most popular approaches to data mining include symbolic [2], [7], [11] and neural [6], [9], [15] models. Symbolic models are represented as either sets of IF - THEN rules, or decision trees generated through symbolic inductive algorithms [2], [7], [11]. A neural model [6], [9], [15] is represented as an architecture of weighted nodes, each of which incorporates a thresholding function. While, in many cases, neural networks lead to more accurate classification results at the expense of long training time, it is difficult to incorporate available domain knowledge into neural networks [13], [14]. Traditionally, neural networks are not able to articulate knowledge in the form of classification rules [13], [14]; however, there exist techniques for extracting certain types of rules from trained neural networks [6], [9], [15].

The following characteristics of real world data may cause difficulties to neural and/or symbolic learning techniques in data mining:

1. Temporal changes of data: During the database lifetime the contents of database can change. None of the existing symbolic or numerical approaches is able to account for this temporal changes of data, i.e., these classifiers need to be completely retrained as data change, which is undesirable in most real-world situations.
2. Need for data preprocessing: Most databases are not designed for data mining. Some existing approaches [1], [9] need special coding of the data before data mining. This may not be suitable for real-world situations as the user is usually not an expert on the method used for data preprocessing.
3. Use of existing knowledge: Whenever there exist expert knowledge on the database it is advantageous to be able to use it. Feedforward neural network approaches [9] are not able to incorporate the available expert knowledge.
4. Data volume: Million of records exist in many databases. Since the feed-forward neural network [9] starts with a full-sized network, which is then pruned to an optimal size, the initial network can be too large in a large dimensional situation.

Fuzzy neural network (FNN) technique can be used as a bridge between numerical and symbolic data representations. Since fuzzy logic has an affinity with the human knowledge representation, it should become a key component of data mining system. One advantage of using fuzzy logic is that we can express knowledge about a database in a manner which is natural for people to comprehend.

In this paper we present a dynamically constructed FNN which can offer solutions to the abovementioned problems. Our FNN can reduce the computational cost as it starts with a minimal rule base which increases only when new input data requires so, at the same time constantly removing irrelevant inputs, rules, and rules conditions which no longer match the data. Dynamical construction of the network also eliminates the trial-and-error determination of the size of the hidden layer in feedforward neural networks [9] and creates a minimal network, thereby reducing the risk of overfitting. The proposed FNN does not need any preprocessing or postprocessing of the data as required in [1] and [9], which is desirable from the user's point of view. The recurrent structure of the proposed network is able to deal with temporal data changes without the requirement to retrain the classifier in contrast to both decision trees [11] and feedforward neural networks [9]. The presented FNN is able to incorporate *existing* domain knowledge in the form of IF-THEN rules, in contrast to feedforward neural networks [9].

X. Z. Wang *et al* [16] used fuzzy neural network for decision making. Our approach is different from X. Z. Wang *et al* [16] in the following aspects. Firstly, our FNN approach is recurrent on-line method which is bottom-up constructed from scratch, thereby generating only a small number of rules, whereas in [16] the initial number of rules is equal to the number of data tuples, which may be too large for large datasets. Secondly, our FNN is able to eliminate irrelevant inputs, rules, and rule conditions, which is necessary to effectively process a large amount of data. X. Z. Wang *et al* [16] used a confidence factor CF , representing

reliability of the rules, defined as

$$CF = \mu_{T_{C1}} \mu_{F_{C1}} \mu_{T_{H11}} / ER ,$$

where $\mu_{T_{C1}}$, $\mu_{F_{C1}}$, $\mu_{T_{H11}}$ are membership values for variables T_{C1} , F_{C1} , T_{H11} , respectively, and ER is the error. Then a $\lambda - Cut$ value was defined as the threshold for the rule to be worth keeping. In such way the rule base size can be reduced. However, in the example given with a $\lambda - Cut \times 10 = 6.0$, the reduced rule base still has multiple instances of the same rule, which makes the rule base ambiguous. For example, rules 2 and 10 are exactly the same,

IF F_{C1} is *Normal* ($\mu = 0.61$) and T_{C1} is *Normal* ($\mu = 1.00$)

THEN T_{H11} is *Normal* ($\mu = 0.67$) ,

and rules 40 and 47 are different only in membership values, i.e., the rule 40 is

IF F_{C1} is *High* ($\mu = 0.67$) and T_{C1} is *High* ($\mu = 0.19$)

THEN T_{H11} is *Normal* ($\mu = 0.55$) ,

and the rule 47 is

IF F_{C1} is *High* ($\mu = 1.00$) and T_{C1} is *High* ($\mu = 1.00$)

THEN T_{H11} is *Normal* ($\mu = 0.22$) .

Consequently, it is difficult to see which rule is applicable to which case.

The FNN of Khan *et al* [8] is partially recurrent, i.e. the recurrent link is between the hidden and the input layer. As the output of the hidden layer and the network are equivalent only for the case where the network has one hidden node and one output node, the recurrency (current inputs depend on past outputs) has ambiguous meaning. Our FNN is fully recurrent, i.e., the recurrent link is from the output to the input layer, which results in the clear interpretation of the rule base. In addition, the network of Khan *et al* [8] is not self-constructing and it does not have a facility for elimination of irrelevant inputs, rules, and rule conditions, which make it unsuitable for large scale problems.

2 Fuzzy Neural Network Structure and Learning Algorithm

The structure of the proposed FNN is shown in Fig. 1. The network consists of four layers, i.e., the input layer, the input membership functions layer, the rule layer, and the output layer. The input nodes represent input variables consisting of the current inputs and the previous outputs. This recurrent structure provides the possibility to include temporal information, i.e., the network learns dynamic input-output mapping instead of static mapping as is feedforward neural networks [9]. It also speeds up the convergence of the network. In databases, data fields are either numerical or categorical. The input membership functions layer

generates input membership functions for numerical inputs, i.e., numerical data values are converted to categorical values. For example, the numerical values of *age* are converted to categorical values of *young*, *middle-aged*, and *old*. Each input node is connected to all membership function nodes for this input. We used piecewise linear-triangular membership functions for computational efficiency. The leftmost and rightmost membership functions are shouldered. Rule nodes are connected to all input membership function nodes and output nodes for this rule. Each rule node performs product of its inputs. The input membership functions act as fuzzy weights between the input layer and the rule layer. Links between the rule layer, the output layer and the input membership functions are adaptive during learning. In the output layer each node receives inputs from all rule nodes connected to this output node and produces the actual output of the system.

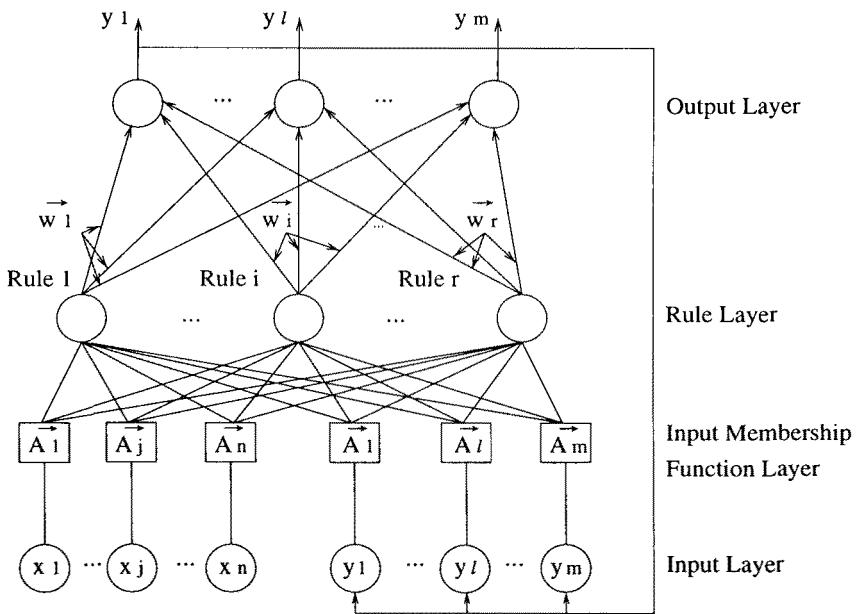


Fig. 1. The structure of our fuzzy neural network

The FNN structure-generation and learning algorithm is as follows:

0. Start with the number of input nodes equal to the sum of the input variables consisting of the current inputs and the past outputs ($n+m$), and the number of output nodes equal to the number of output variables (m). The rule layer is empty, i.e., there are initially no rules in the rule base;
1. Add two equally spaced input membership functions along the operating range of each input variable. In such a way these membership functions will

satisfy ϵ -completeness, which means that for a given value of x of one of the inputs in the operating range, we can always find a linguistic label A such that $\mu_A(x) \geq \epsilon$. If the ϵ -completeness is not satisfied, there may be no rule applicable for a new data input;

2. Create initial rule base layer using the following form for rule i :

$$\begin{aligned} \text{Rule } i: \quad & \text{IF } x_1 \text{ is } A_1^i \text{ and } \dots \text{ } x_n \text{ is } A_n^i \\ & \text{and } y_1(k-1) \text{ is } A_1^i \text{ and } \dots \text{ } y_m(k-r) \text{ is } A_m^i \\ & \text{THEN } y_l = w_l^i, \dots, y_m = w_m^i, \end{aligned} \quad (1)$$

where x_j and y_l ($l = 1, 2, \dots, m$) are the current inputs and the past outputs, respectively, w_l^i is a real number. Here A_j^i is the membership function of the antecedent part of the i rule for the j input node, k is the time, and r is the delay. The membership value μ_i of the premise of the i th rule is calculated as fuzzy AND using the product operator

$$\mu_i = A_1^i(x_1) \cdot A_2^i(x_2) \cdot \dots \cdot A_n^i(x_n). \quad (2)$$

The output y_l of the fuzzy inference is obtained using weighted average

$$y_l = \frac{\sum_i \mu_i w_l^i}{\sum_i \mu_i}; \quad (3)$$

3. Train the network using the following learning rules

$$w_l^i(k+1) = w_l^i(k) - \eta \frac{\partial \varepsilon_l}{\partial w_l^i}. \quad (4)$$

$$A_j^i(k+1) = A_j^i(k) - \eta \frac{\partial \varepsilon_l}{\partial A_j^i}, \quad (5)$$

where η is the learning rate. The objective is to minimize an error function

$$\varepsilon_l = \frac{1}{2}(y_l - y_{dl})^2, \quad (6)$$

where y_l is the current output, y_{dl} is the target output. The learning rate η is adaptive to improve the speed of convergence, as well as the learning performance (accuracy). We start with a basic learning rate to enhance the learning speed. Whenever ε_l changes its sign, the learning rate is reduced according to the following iterative formula

$$\eta_{\text{new}} = rc \eta_{\text{old}}, \quad (7)$$

where rc is a coefficient in the range $(0, 1)$. The learning error ε_l asymptotically approaches zero or a pre-specified small value > 0 as the iteration number k increases;

4. If the degree of overlapping of membership functions is greater than a threshold (e.g. 0.9), combine those membership functions. We use the following *fuzzy similarity measure* [4]

$$\text{Degree}(A_1 = A_2) = E(A_1, A_2) = \frac{M(A_1 \cap A_2)}{M(A_1 \cup A_2)}, \quad (8)$$

where \cap and \cup denote the intersection and the union of two fuzzy sets A_1 and A_2 , respectively. $M(\cdot)$ is the size of a fuzzy set, and $0 \leq E(A_1, A_2) \leq 1$. If an input variable ends up with only one membership function, which means that this input is irrelevant, delete the input. We can thus eliminate irrelevant inputs and reduce the size of the rule base. If the classification accuracy of FNN is below the requirement (e.g. 99%), and the number of rules is less than the specified maximum, go to step 6. Otherwise, go to step 5;

5. The generated rules are evaluated for accuracy and generality. We use a weighting parameter between accuracy and generality, the rule applicability coefficient (weighting of the rules) (WR) which is defined as the product of the number of the rule activations RA by the accuracy of the rule A in terms of misclassifications. All rules whose rule applicability coefficient WR falls below the defined threshold (e.g. 10) are deleted. Elimination of rule nodes is *rule by rule*, i.e., when a rule node is deleted, associated input membership nodes and links are deleted as well. By varying WR threshold a user is able to specify the degree of rule base compactness. The size of the rule base can thus be kept minimal. If the classification accuracy of the FNN after pruning is below the requirement (e.g. 90%), go to step 6, otherwise stop;
6. Add additional membership function for each input at its value at the point of the maximum output misclassification error. One vertex of additional membership function is placed at the value at the point of the maximum output error and has membership value unity; the other two vertices lie at the centers of the two neighbouring regions, respectively, and have membership values zero. As the output of the network is not a binary 0 or 1, but a continuous function in the range from 0 to 1, by firstly eliminating the errors whose deviation from the target values is the greatest, we can speed up the convergence of the network substantially;
7. Update the rule base layer in the same way as in step 2;
8. Retrain the network with the updated rule base layer using the algorithm given in step 3;
9. Go to step 4.

3 Experimental Results

To test our algorithm we used ten classification problems of different complexity defined in [1] on synthetic database with nine attributes given in Table 1. Attributes *elevel*, *car* and *zipcode* are categorical, and all others are non-categorical. Functions 1 to 5 have predicates with one (function 1), two (functions 2 and 4), and three attribute values (functions 4, 5, and 6). Functions 7 to 9 are linear

functions and function 10 is a non-linear function of attribute values. Consult Agrawal *et al* [1] for detailed description of the database and the functions.

Table 1. Description of attributes adapted from [1]

Attribute	Description	Value
<i>salary</i>	salary	uniformly distributed from 20K to 150K
<i>commission</i>	commission	$\text{salary} \geq 75K \Rightarrow \text{commission} = 0 \text{ else}$
		uniformly distributed from 10k to 75K
<i>age</i>	age	uniformly distributed from 20 to 80
<i>elevel</i>	education level	uniformly chosen from 0 to 4
<i>car</i>	make of the car	uniformly chosen from 1 to 20
<i>zipcode</i>	zip code of the town	uniformly chosen from 9 available zipcodes
<i>hvalue</i>	value of the house	uniformly distributed from n50K to n150K where $n \in 0 \dots 9$ depends on zipcode
<i>hyers</i>	years house owned	uniformly distributed from 1 to 30
<i>loan</i>	total loan amount	uniformly distributed from 0 to 500K

Attribute values were randomly generated according to uniform distribution as in [1]. For each experiment we generated 1000 training and 1000 test data tuples. As the tuples were classified into two classes only (Groups A and B), a single output node was sufficient. Default class was Group B. The target output was 1 if the tuple belongs to Group A, and 0 otherwise. We used the random data generator with the same perturbation factor $p = 5\%$ as in [1] to model fuzzy boundaries between classes. For comparison with our approach we used decision trees algorithms C4.5 and C4.5rules [12] on the same data sets. We also compared our results with those of a pruned feedforward neural network (NeuroRule) of Lu *et al* reported in [9] for the same classification problems. Table 2 shows the accuracy on the test data set, the number of rules, and the average number of conditions per rule, for all three approaches, with two sets of FNN parameters. We report only the rules for class A for C4.5 to make the comparison more objective, as C4.5 generates rules for both A and B classes, while both our FNN and NeuroRule [9] generate rules only for class A, using class B as a default rule.

With the first set of parameters in Table 2 (indicated by FN1) and compared to NeuroRule, our FNN produces the rule bases of less complexity for all functions, except 7 and 9. The FNN gives better accuracy for functions 6, 7, and 9, the same for function 1, and worse for the rest. Compared to C.4.5, the FNN gives less complex rule bases for functions 2, 4, and 9, the same for functions 2, 4, and 7, and more complex for the rest. Our FNN gives better accuracy than C4.5 on function 8, 9, and 10, similar for function 1, 6, and 7 and worse for the rest. The results for functions 8 and 10 were not reported in [9] (indicated by "N/A" in Table 2) as these functions lead to highly skewed data.

Table 2. Accuracy rates on the test data set, the number of rules, and the average number of conditions per rule for NeuroRule (NR) (performance data taken from [9]), the C4.5, and the FNN (FN). FN1 uses the following parameters: the learning rate $\eta = 0.0001$, the coefficient for learning rate adaptation $rc = 0.9$, required classification accuracy for training 99%, required classification accuracy after pruning 90%, maximum number of rules = 20, degree of overlapping of membership functions = 0.9, weighting of the rules $WR = 10$, maximum number of iterations $k = 10$. FN2 uses the same parameters as FN1, except weighting of the rules $WR = 15$

Func.	Accuracy				No. of Rules				No. of Conditions			
	NR	C4.5	FN1	FN2	NR	C4.5	FN1	FN2	NR	C4.5	FN1	FN2
1	99.91	99.9	99.9	100	2.03	2	2	3	2.23	1	1	1
2	98.13	99.4	96.8	93.3	7.13	3	3	4	4.37	3.33	2.33	2
3	98.18	99.8	97.5	100	6.70	5	5	6	3.18	2	2	2
4	95.45	99.3	95.0	95.3	13.37	11	7	8	4.17	4.18	3.14	3.13
5	97.16	98.6	96.4	93.0	24.40	4	5	7	4.68	4.75	4.2	4
6	90.78	95.8	94.7	95.1	13.13	8	9	10	4.61	3.37	3	3.1
7	90.50	95.4	95.2	95.9	7.43	12	10	13	2.94	2.17	2.9	2.54
8	N/A	98.1	99.1	98.9	N/A	2	3	4	N/A	1	2.67	2.75
9	90.96	92.6	96.7	97.2	9.03	12	9	11	3.46	2.75	3.56	3.64
10	N/A	95.5	96.0	97.0	N/A	6	8	9	N/A	2.17	4.5	4.67

In Table 2 for FN1 we attempted to obtain the most compact rule base to allow for easy analysis of the rules for very large databases, and thus easy decision making in real-world situations. If accuracy is more important than compactness of the rule base, it is possible to use our FNN with more strict accuracy requirement, i.e., a higher threshold for pruning the rule base (WR), thereby producing more accurate rules at the expense of rule base complexity. The final decision regarding complexity versus accuracy of the rules is application specific. Table 2 shows the performance comparison with a weaker compactness constrain, i.e., a larger WR , for FN2. In this case the FNN offers similar or better accuracy compared to both C4.5 and NeuroRule, except for functions 2, 4, and 5. Comparing the performance of FN2 with FN1, we see that relaxing the compactness constrain (increasing WR) improves accuracy for functions 1, 3, 4, 6, 7, 9, and 10. For functions 2, 5, and 8, however, relaxing the compactness constrain actually results in lower accuracy, which means that caution is necessary in selecting appropriate values of the weighting of the rules WR . As all problems in the table are different, the different compactness constrain values for each function may be needed. We used the same WR value for all functions to make a balanced comparison.

4 Conclusion and Discussion

In this paper we presented a dynamically-constructed recurrent fuzzy neural network for knowledge discovery from databases. Experiments were conducted to test the proposed approach to a well defined set of data mining problems given by [1]. The results shows that the presented approach is able to achieve accuracy and compactness comparable to both feedforward neural networks and decision tree methods, with more compact rule base than feedforward neural network for most of datasets used, and for some compared to decision tree approach. The proposed method is also able to achieve a higher accuracy on some datasets compared to both feedforward neural network and decision tree approaches. Incremental on-line learning of the proposed approach requires less time compared to off-line training of feedforward neural network approaches [9] due to the local updating feature of fuzzy logic, i.e., our FNN only updates the rules applicable to a current data while feedforward neural network globally updates the network. The proposed method eliminates extraction of symbolic rules phase in both decision trees and feedforward neural networks approaches. Our FNN permits updating of the rules along with changes in database contents due to its recurrent structure, in contrast to the decision tree and the feedforward neural networks methods.

Another important feature of databases need to be considered. Data may contain a certain level of noise due to statistical fluctuations or human errors. Using neural network based approach such as [9] and the one proposed in this paper should also greatly reduce the problem of data noise, due to ability of neural networks to deal effectively with noisy data. Testing of the proposed FNN in this aspect is subject of a future work.

References

1. Agrawal R., Imielinski T., and Swami A.: Database Mining: A Performance Perspective. *IEEE Trans. Knowledge and Data Engineering* **5** (1993) 914–925
2. Breiman L., Friedman J. H., Olshen R. A., and Stone C. J.: Classification and Regression Trees: Wansworth International (1984)
3. Cercone N. and Tsuchiya M.: Special Issue on Learning and Discovery in Knowledge-based Databases. *IEEE Trans. Knowledge and Data Engineering* **5** (1993)
4. Dubois D. and Prade H.: A Unifying View of Comparison Indices in a Fuzzy Set Theoretic Framework. in Yager R. R. (ed.) *Fuzzy Sets and Possibility Theory: Recent Developments*: Pergamon NY (1982)
5. Frawley W. J., Piatetsky-Shapiro G., and Matheus C. J.: Knowledge Discovery in Databases: An Overview. In Piatetsky-Shapiro G. and Frawley W. J. (eds.): *Knowledge discovery in databases*: AAAI Press/MIT Press (1991) 1–27
6. Gallant S. I.: Connectionist Expert Systems. *Communications of the ACM* **32** (1988) 153–168
7. Kerber R.: Learning Classification Rules from Examples. Proc. 1991 AAAI Workshop on Knowledge Discovery in Databases: AAAI (1991)
8. Khan E. and Unal F.: Recurrent Fuzzy Logic Using Neural Networks. Proc. 1994 IEEE Nagoya World Wisepersons Workshop (1994) 48–55

9. Lu H., Setiono R., and Liu H.: Effective Data Mining Using Neural Networks. *IEEE Trans. on Knowledge and Data Engineering* **8** (1996) 957–961
10. Piatetsky-Shapiro G.: Special Issue on Knowledge Discovery in Databases - from Research to Applications. *Int. J. of Intelligent Systems* **5** (1995)
11. Quinlan J. R.: Induction of Decision Trees. *Machine Learning* **1** (1986) 81–106
12. Quinlan J. R.: C4.5:Programs for Machine Learning Morgan Kaufmann: San Mateo CA (1993)
13. Quinlan J. R.: Comparing Connectionist and Symbolic Learning Methods. In S. Hanson, G. Drastall, and R. Rivest (eds.): *Computational Learning Theory and Natural Learning Systems*: MIT Press **1** (1994) 445–456
14. Shavlik J. W., Mooney R. J., and Towell G. G.: Symbolic and Neural Learning Algorithms: An Experimental Comparison. *Machine Learning* **6** (1991) 111–143
15. Towell G. G. and Shavlik J. W.: Extracting Refined Rules From Knowledge-based Neural Networks. *Machine Learning* **13** (1993) 71–101
16. Wang X. Z., Chen B. H., Yang S. H., McGreavy C., Lu M. L.: Fuzzy Rule Generation From Data for Process Operational Decision Support. *Computer and Chemical Engineering* **21** (1997) 661–666
17. Wu X.: Knowledge Acquisition from Databases. Ablex Publishing: Norwood NJ (1995)

CCAIIA: Clustering Categorical Attributes into Interesting Association Rules

Brett Gray and M E Orlowska

School of Information Technology, University of Queensland, QLD 4072 Australia,
brett@psych.uq.edu.au, maria@dstc.edu.au

Abstract. We investigate the problem of mining interesting association rules over a pair of categorical attributes at any level of data granularity. We do this by integrating the rule discovery process with a form of clustering. This allows associations between groups of items to be formed where the grouping of items is based on maximising the “interestingness” of the associations discovered. Previous work on mining generalised associations assumes either a distance metric on the attribute values or a taxonomy over the items mined. These methods use the metric/taxonomy to limit the space of possible associations that can be found. We develop a measure of the interestingness of association rules based on support and the dependency between the item sets and use this measure to guide the search. We apply the method to a data set and observe the extraction of “interesting” associations. This method could allow interesting and unexpected associations to be discovered as the search space is not being limited by user defined hierarchies.

1 Introduction

Data mining and knowledge discovery in databases has emerged as a new area of research, attracting significant attention from the database and machine learning communities (See [4] for an overview). An interesting sub field is the problem of discovering association rules.

1.1 Association Rules

Initially introduced in [1], the original motivation was to analyse supermarket transactions and observe how often items are purchased together. Given a set of transactions, each representing a set of purchased items, an association rule is a rule of the form $X \Rightarrow Y$ where X and Y are both sets of items. The rule has support $s\%$, and confidence $c\%: s\%$ of all transactions contain all items from $X \cup Y$ and $c\%$ of all transactions containing all items from X also contain all items from Y . The problem as introduced in [1] is to mine all association rules that achieve a minimum support minsup , and confidence minconf with minsup and minconf being user specified parameters. This original formulation of the problem has undergone significant research to develop more efficient algorithms for extracting rules [2], [14].

This supermarket transaction formulation can be viewed as discovering associations over a large relational table of boolean attributes. Each tuple corresponds to a transaction and each boolean attribute corresponds to a supermarket item, adopting a value 1 if the item is present in the transaction or 0 otherwise. Further research has extended the problem to include quantitative attributes [13], [7] which have some natural notion of a distance measure (eg. age, income) and categorical attributes [13] containing a range of values with no obvious distance measure (eg. zip code, customer id).

A problem that has been identified with mining association rules is that of granularity [3], [12]. Consider the supermarket transaction case described above. While the association $\{\text{Milk}\} \Rightarrow \{\text{Bread}\}$ may be an association that achieves the minimum support and confidence, if the data is represented at a different level of granularity, this association will not be found. The database may contain items such as brand X skim milk, brand Y full cream milk, brand Z multigrain bread, etc. At this finer level of granularity there may not be any association (eg. $\{\text{brand X skim milk}\} \Rightarrow \{\text{brand Z multigrain bread}\}$) that achieves minimum support or confidence and the $\{\text{Milk}\} \Rightarrow \{\text{Bread}\}$ association will remain undiscovered.

The problem of granularity has been addressed on quantitative attributes by allowing associations between ranges of values where an attributes value must lie within a range to conform to the association [13]. Work has also been done on mining associations where the left hand side is restricted to two quantitative attributes and the right hand side to a particular value of a boolean attribute [7]. The left hand side then adopts a rectangle or an admissible region (connected, x-monotone region). For a tuple to conform to the association the pair of quantitative attributes must be within the defined region and the boolean attribute must adopt the specified value.

The granularity problem has been addressed on sets of boolean attributes (eg. supermarket transaction data) by assuming a taxonomy (is-a hierarchy) over the attributes [12], [9]. Associations can then be mined between items and nodes of the taxonomy. For example, a taxonomy could be constructed which grouped all forms of milk to a milk node and all forms of bread to a bread node. The $\{\text{Milk}\} \Rightarrow \{\text{Bread}\}$ association could then be mined as a generalised association between the milk and bread nodes of the taxonomy. It is easy to see how this method could be applied to categorical attributes with a taxonomy over the values of the attribute.

The use of a taxonomy to mine generalised association rules has the benefit of providing a method for incorporating additional domain information into the mining process, in the form of the taxonomy. However it is also limiting in that it not only requires the construction of the taxonomy but it also limits the forms of associations that can be discovered. Consider an example of mining generalised associations between a pair of categorical attributes, Customer Id and Item Code. Each tuple represents the identification of a customer and an item that customer has purchased at a particular store. We wish to mine generalised association rules in order to analyse customer profiles. Particular groups of customers may

typically purchase particular subsets of items (eg. low income earners may buy a lot of products that are the cheapest in their range). These customer profiles could then help in devising marketing strategies. Generalised associations could potentially be mined by constructing a taxonomy over the Customer Id and Item Code attributes that forms groups of customers based on income and groups of items based on cost relative to other similar items. Generalised associations such as {low income earners} \Rightarrow {low cost products} could then be discovered as associations between taxonomy nodes, however associations such as {busy people} \Rightarrow {convenience foods} and {people who cook} \Rightarrow {cooking foods} would never be found as the {busy people}, {convenience foods}, {people who cook} and {cooking foods} groupings are not contained in the taxonomy.

This limitation taxonomies place on discovering rules could prove to be a greater curse if the taxonomy is constructed purely for the purpose of mining associations. In this case the construction of the taxonomy could be biased by the forms of rules that are expected to arise. Unexpected rules, which could potentially be the most interesting and beneficial rules to discover would be less likely to occur in the taxonomy and therefore more likely to remain undiscovered.

In order to address this limitation taxonomies provide on mining associations over categorical attributes, we define a measure of the interestingness of association rules and then present the algorithm CCAIIA (Clustering Categorical Attributes Into Interesting Association rules). CCAIIA extracts interesting association rules between sets of items from a pair of categorical attributes. The algorithm does not assume the existence of a taxonomy over the attributes, but groups attribute values by a form of clustering which attempts to maximise the interestingness of the discovered rules.

1.2 Clustering Algorithms

Cluster analysis is the classification of items into a number of groups, or clusters, and has attracted a lot of attention in the data mining literature [11], [5], [4]. The grouping of items is generally designed to heuristically minimise some criteria. Algorithms such as BIRCH [15], assume a distance measure on the values to be clustered and form groupings that minimise measurements such as the average inter-cluster distance, or the centroid Euclidean distance. DBSCAN [5] is an algorithm that has been applied to spacial data and finds clusters of points that are above a certain density. Conceptual clustering algorithms such as COBWEB [6] do not assume a distance measure on the attributes and can therefore be applied to categorical attributes. COBWEB [6] forms clusters based on a category utility measure which indicates the increase in attribute-value predictiveness given the clustering, compared to a lack of clustering. CLUSTER/2 [10] performs conceptual clustering which produces clusters that can be easily characterised by simple, well fitting conjunctive statements.

- Clustering algorithms are typically one of two forms: hierarchical or partitioning. Hierarchical algorithms form a hierarchical grouping of the data set by repeatedly joining (bottom up) or dividing (top down) existing clusters into new clusters. Partitioning algorithms partition the data set into a specified number

of clusters by iteratively selecting a representative point for each cluster, determining a partitioning of the data based on these representative points and evaluating the partitioning according to some criteria.

Our approach integrates the clustering process with the discovery of association rules. CCAIIA performs a form of bottom up hierarchical clustering which heuristically maximises the interestingness of the rules discovered.

1.3 Paper Outline

The remainder of this paper is organised as follows: Section 2 introduces our measure of the interestingness of an association rule and formalises the problem. Section 3 outlines the CCAIIA algorithm. Section 4 presents the application of the algorithm to an artificially generated data set and section 5 gives some concluding remarks.

2 Deriving a Measure of the Interestingness of Association Rules

We will start by formalising a definition of generalised association rules over a pair of categorical attributes with the associated support and confidence measures, and then provide additional measures in order to facilitate the clustering of attributes values.

Let X and Y be two categorical attributes with $\text{Dom}(X) = \{x_1, \dots, x_k\}$ and $\text{Dom}(Y) = \{y_1, \dots, y_p\}$. Let $r = \{t_1, \dots, t_n\}$ be a set of tuples such that for each tuple, t , we have $t(X) \in \text{Dom}(X)$ and $t(Y) \in \text{Dom}(Y)$.

For any attribute $W \in \{X, Y\}$ and any set, $W_1 \subseteq \text{Dom}(W)$, let the support of W_1 in r ($s(W_1, r)$) be the percentage of tuples (t) in r for which $t(W)$ is in the set W_1 :

$$s(W_1, r) = \#\{t \in r \mid t(W) \in W_1\} / \#r \times 100\%$$

Let $X_1 \subseteq \text{Dom}(X)$ and $Y_1 \subseteq \text{Dom}(Y)$. $X_1 \Rightarrow Y_1$ is then a generalised association rule with support and confidence defined as follows:

$$s(X_1 \Rightarrow Y_1, r) = \#\{t \in r \mid t(X) \in X_1 \wedge t(Y) \in Y_1\} / \#r \times 100\%$$

$$c(X_1 \Rightarrow Y_1, r) = s(X_1 \Rightarrow Y_1, r) / s(X_1, r) \times 100\%$$

Confidence refers to the percentage of tuples that support X_1 which also support Y_1 .

Note: We adopt the notation $s(W_1)$, $s(X_1 \Rightarrow Y_1)$ and $c(X_1 \Rightarrow Y_1)$ for $s(W_1, r)$, $s(X_1 \Rightarrow Y_1, r)$ and $c(X_1 \Rightarrow Y_1, r)$ respectively when r is obvious from the context.

Alternative notation: Let $P(X_1)$ be the probability that for a random tuple t from r , $t(X) \in X_1$. Similarly let $P(Y_1)$ be the probability that $t(Y) \in Y_1$. Then $s(X_1) = P(X_1)$, $s(Y_1) = P(Y_1)$, $s(X_1 \Rightarrow Y_1) = P(X_1 \cap Y_1)$ and $c(X_1 \Rightarrow Y_1) = P(X_1 \cap Y_1) / P(X_1) = P(Y_1 | X_1)$.

2.1 Why Support and Confidence are not Sufficient

While support and confidence provide a useful characterisation of association rules, they are not sufficient to derive an interestingness measure or a heuristic for clustering. Consider the Customer Id, Item Code example, and a generalised association rule $\{\text{low income earners}\} \Rightarrow \{\text{low fat products}\}$ with support 10% and confidence 40%. While this rule has achieved a strong support and confidence, it would not be very interesting if 40% of all purchases were low fat products. If this were the case then the rule is not discriminating low income earners from customers in general. Additionally, support and confidence are not sufficient to derive a clustering heuristic. If the clustering procedure was designed to maximise support and confidence of discovered rules, then the clusters $X_1 = \text{Dom}(X)$ and $Y_1 = \text{Dom}(Y)$ with the rule $X_1 \Rightarrow Y_1$ would achieve 100% support and confidence, although this rule would not prove very interesting. For these reasons we define the measures discrimination and interestingness.

2.2 Discrimination and Interestingness

Let the discrimination (d) of the generalised association rule $X_1 \Rightarrow Y_1$ be the following:

$$d(X_1 \Rightarrow Y_1, r) = c(X_1 \Rightarrow Y_1)/s(Y_1)$$

Alternativley,

$$d(X_1 \Rightarrow Y_1, r) = \frac{P(X_1 \cap Y_1)}{P(X_1) \times P(Y_1)}$$

This measure gives an indication of the independence of X_1 and Y_1 (or the independence of the events that a random tuple supports X_1 and a random tuple supports Y_1). A value of 1 indicates that X_1 and Y_1 are independent. A value greater than 1 indicates a stronger relationship between the events than would be expected if the events were independent, and Y_1 therefore discriminates X_1 from $\text{Dom}(X)$. A value between 0 and 1 indicates a relationship between the events that is weaker than independence.

We consider an interesting rule to be a rule which obtains a high support and discrimination. In order to express this in a single measure we define interestingness (i) as

$$i(X_1 \Rightarrow Y_1, r) = (d(X_1 \Rightarrow Y_1, r)^l - 1) \times s(X_1 \Rightarrow Y_1)^m$$

l and m are parameters to weight the relative importance of the two measures. When l and m are greater than 0 and $d(X_1 \Rightarrow Y_1, r) > 1$, $i(X_1 \Rightarrow Y_1, r)$ is monotonically increasing with both $d(X_1 \Rightarrow Y_1, r)$ and $s(X_1 \Rightarrow Y_1)$. A support of 0 or a discrimination of 1 (indicating independence of X_1 and Y_1) centers the function at 0.

Note: We adopt the notation $d(X_1 \Rightarrow Y_1)$ and $i(X_1 \Rightarrow Y_1)$ for $d(X_1 \Rightarrow Y_1, r)$ and $i(X_1 \Rightarrow Y_1, r)$ respectivley when r is obvious from the context.

2.3 Problem Formulation

The problem addressed in this paper is given two categorical attributes X and Y , we wish to mine a limited number of generalised association rules $X_1 \Rightarrow Y_1$, where $X_1 \subseteq \text{Dom}(X)$ and $Y_1 \subseteq \text{Dom}(Y)$. These association rules should be formed in a way that heuristically maximises the interestingness of the associations discovered.

3 CCAIIA Algorithm

3.1 Overview

The algorithm is a form of bottom up hierarchical clustering where initially each different $x \in \text{Dom}(X)$ value present in the data set is a cluster. The algorithm then iteratively merges pairs of clusters into single clusters by forming the union of the set of values in each of the two clusters being merged. This process of iteratively merging pairs of clusters continues until a termination criteria is met.

For each cluster, C_i , a list, $C_i\text{-list}$, of $(y, \text{count}(y, C_i, r))$ pairs is maintained where $y \in \text{Dom}(Y)$, and

$$\text{count}(y, C_i, r) = \#\{t \in r \mid t(X) \in C_i \wedge t(Y) = y\}$$

The list contains all y values with $\text{count}(y, C_i, r) \geq 1$. We will call the set of all such y values $RHS'(C_i, r)$ or RHS' if C_i and r are both obvious from the context.

Given a cluster C_i , and its associated $C_i\text{-list}$, RHS' can be pruned to form $RHS \subseteq RHS'$. This then gives an association rule:

$$(\text{Rule } C_i) : C_i \Rightarrow RHS$$

The pruning of RHS' is called RHS' pruning and attempts to maximise the interestingness of Rule C_i .

In addition to this, a hash table is maintained, storing the support of each value $y \in \text{Dom}(Y)$ that is present in the data set. This facilitates the calculation of discrimination values for rules.

The structure of the algorithm is as follows:

- (1) Establish the hash table of $y \in \text{Dom}(Y)$ support values.
- (2) Establish the initial set of clusters C , $C = \{C_1, C_2, \dots, C_k\}$ where each cluster $C_i = \{x_i\}$, $i = 1..k$, $x_i \in \text{Dom}(X)$ and occurs in data set.
For each C_i , $i = 1..k$, establish $C_i\text{-list}$ and perform RHS' pruning.

repeat

- (3) Select a pair of clusters, $C_i \in C$ and $C_j \in C$ to merge.
- (4) if Rule C_i passes rule pruning criteria then report Rule C_i .

- (5) if Rule C_j passes rule pruning criteria then report Rule C_j .
 - (6) Merge C_i and C_j into C_i :
 $C_i := C_i \cup C_j$
Merge C_j -list into C_i -list
Perform RHS' pruning on C_i
 - (7) $C := C - C_j$
- until** (termination criteria is meet or $\#C = 1$)
forall clusters $C_i \in C$ **do**
- (8) if Rule C_i passes rule pruning criteria then report Rule C_i .

3.2 RHS' Pruning

RHS' pruning is completed by initially starting with $RHS = \{\}$ and then incrementally adding the item $y_i \in RHS' - RHS$ with the highest discrimination of all items in $RHS' - RHS$. The discrimination of an item $y \in \text{Dom}(Y)$ is equal to $d(C_i \Rightarrow \{y\})$. This process continues until either $RHS' - RHS = \{\}$ or adding the next item would cause $i(C_i \Rightarrow RHS)$ to lower.

While this procedure is not guaranteed to find the subset that is optimal for i , we adopt it as a reasonable approximation due to the following property:

Lemma: At all points of the procedure when $RHS \neq \{\}$, the rule $C_i \Rightarrow RHS$ achieves the largest possible discrimination of all rules with at least the same support in which the right hand side is a subset of RHS' .

The proof can be found in an associated technical report [8].

3.3 Selection of Clusters to Merge and Termination Criteria

To select a pair of clusters to merge, a measure is used to give an indication of the gain in interestingness that will be achieved over the interestingness if the RHS sets of the clusters being merged were independent.

For each pair of clusters C_i and C_j , we have the value

$$gain = \frac{i(C_i \cup C_j \Rightarrow RHS_i \cup RHS_j)}{\text{indepi}(s(C_i), s(C_j), s(RHS_i), s(RHS_j), c(C_i \Rightarrow RHS_i), c(C_j \Rightarrow RHS_j))}$$

RHS_i and RHS_j are the sets RHS for C_i and C_j respectively. indepi is a function that calculates $i(C_i \cup C_j \Rightarrow RHS_i \cup RHS_j)$ in terms of $s(C_i)$, $s(C_j)$, $s(RHS_i)$, $s(RHS_j)$, $c(C_i \Rightarrow RHS_i)$ and $c(C_j \Rightarrow RHS_j)$ from the assumption that RHS_j is independent of C_i and RHS_i and that RHS_i is independent of C_j and RHS_j . This function is fully expanded in the associated technical report [8].

The pair of clusters selected is the pair with the highest gain value. The gain value also provides a termination criteria. When no two clusters in C have a gain value above the user specified parameter $termgain$ then the termination criteria is met.

3.4 Rule Pruning

Even for moderate sizes of $\text{Dom}(X)$, the number of rules generated from the algorithm can be large. For this reason we apply a number of rule pruning strategies to retrieve the minimum amount of interesting rules.

As for the original association rule formulation, the parameters minsup and minconf can be used to prune rules without a suitable level of support or confidence. We have also applied optional minimum parameters to the discrimination and interesting measures, mindis and minint .

As well as this, we also prune rules relative to their parent and children. The parameter mingain refers to the minimum level of gain calculated in forming the cluster from its children, providing the cluster has children. This implies the rule must be sufficiently interesting with respect to its children's rules in order to pass. We also adopt a parameter pargain , in which, if a cluster has a parent, it must pass the test

$$\text{gain1} \geq \text{gain2}/\text{pargain}$$

where gain1 is the gain calculated in forming the rule and gain2 is the gain calculated when merging the rule into a parent cluster. This implies the rule must be sufficiently interesting with respect to its parent in order to survive.

4 Application to Data Set

We present the results of applying the algorithm to an artificially generated data set representing customer purchase patterns. We generate the data to contain three associations with reasonable support and discrimination, and then analyse the associations discovered by CCAIIA in terms of the generated associations.

4.1 Data Set Generation and Parameter Settings

The data set consists of two attributes, Customer Id and Item Code and we wish to mine generalised associations from the Customer Id attribute to the Item Code attribute. We assume that $\#\text{Dom}(\text{CustomerId}) = 1000$ and label the customer identifications $x_1 - x_{1000}$ (ie. $\text{Dom}(\text{Customer Id}) = \{x_1, x_2, \dots, x_{1000}\}$). Similarly we define $\text{Dom}(\text{Item Code}) = \{y_1, y_2, \dots, y_{1000}\}$.

We generated the data to contain the associations:

- (Rule 1) : $\{\text{busy people}\} \Rightarrow \{\text{convenience foods}\}$ s = 12% c = 60%
- (Rule 2) : $\{\text{people who cook}\} \Rightarrow \{\text{cooking foods}\}$ s = 24% c = 80%
- (Rule 3) : $\{\text{low income earners}\} \Rightarrow \{\text{low cost products}\}$ s = 21% c = 70%

The sets adopted in the rules are defined as follows: $\{\text{busy people}\} = \{x_1, \dots, x_{200}\}$, $\{\text{people who cook}\} = \{x_{201}, \dots, x_{500}\}$, $\{\text{low income earners}\} = \{x_{501}, \dots, x_{800}\}$, $\{\text{convenience foods}\} = \{y_1, \dots, y_{200}\}$, $\{\text{cooking foods}\} = \{y_{201}, \dots, y_{600}\}$, $\{\text{low cost products}\} = \{y_{xx1}, y_{xx2}, y_{xx3}\}$.

Where xxi is any number with the last digit as i . Full details of the data set generation procedure can be found in the associated technical report [8].

The parameters of the algorithm were determined through experimentation and set to the following values:

$l = 2, m = 1, \text{minsup} = 10\%, \text{minconf} = \text{unused}, \text{mindis} = 1.5, \text{minint} = \text{unused}, \text{mingain} = 1.09, \text{pargain} = 0.996, \text{termgain} = 1.05$

4.2 Results

The application of the algorithm to the data set resulted in the discovery of 8 rules, representing the 3 generated rules to varying levels of completeness. To present each of the discovered rules, we calculate which of the generated rules is most strongly represented. This is done by calculating the percentage of values in the left hand side (antecedent) of the discovered rule that belongs to each of the three generated rules, and adopting the generated rule with the largest value.

Tables 1 and 2 present statistics on how closely each of the discovered rules represents its generated rule. The statistics for table 1 are as follows:

1. **%Generated Antecedent** represents the percentage of values in the generated rule's antecedent (left hand side) that are contained in the antecedent of the discovered rule. The larger this value, the more completely the antecedent of the generated rule is represented.
2. **%Antecedent Complement** refers to the percentage of values from the complement of the generated rule's antecedent (ie. Customer Ids not in the left hand side of the generated rule) that are contained in the antecedent of the discovered rule. The smaller this value, the more concise the representation of the generated antecedent.
3. **%Antecedent Correct** refers to the percentage of values from the discovered antecedent that are contained in the generated antecedent. This gives an indication of the strength of the discovered rules association to the corresponding generated rule.

The statistics for table 2 (%Generated Consequent, %Consequent Complement and %Consequent Correct) adopt the same meaning as the corresponding antecedent rules but refer to the rules consequents (right hand sides).

4.3 Discussion

The statistics presented indicates a strong representation of all three generated rules. Due to the random nature of the data generation as well as the overlap between the consequent of the third rule and the first two rules, it is unrealistic to expect perfect rule representation. The greatest benefit of this algorithm (as with many data mining tools in general) for realistic data analysis would be achieved

Table 1. Accuracy of Discovered Antecedent to Generated Antecedent

Discovered Rule Number	Generated Rule Number	%Generated Antecedent	%Antecedent Complement	%Antecedent Correct
1	1	81	7	74.3119
2	1	79.5	6.625	75
3	2	70.3333	4.14286	87.9167
4	2	66.6667	2.28571	92.5926
5	2	60.6667	2	92.8571
6	2	58	1.71429	93.5484
7	3	70.3333	3.42857	89.7872
8	3	63	2.28571	92.1951

Table 2. Accuracy of Discovered Consequent to Generated Consequent

Discovered Rule Number	Generated Rule Number	%Generated Consequent	%Consequent Complement	%Consequent Correct
1	1	84	3.625	85.2792
2	1	87	3.75	89.2941
3	2	68	2.33333	95.1049
4	2	67.5	1.66667	96.4286
5	2	66.5	2.16667	95.3405
6	2	66.75	2	95.6989
7	3	77.3333	3.57143	90.2724
8	3	73.6667	4.71429	87.0079

when used in conjunction with a domain specific expert, who could look at the clusters formed and identify the common qualities of the grouped items (eg. the fact that 96% of the items are cooking foods). The results indicate that the discovered rules are of a strength that would allow these trends to be identified. The comparably strong results for both the consequent and the antecedent indicates that the devised interestingness measure is providing a strong basis for both the RHS' pruning procedure and the selection of clusters to merge. As well as this the rule pruning procedure has lowered the number of generated rules from a potential of 1999 to 8, making the quantity of returned data easily accessible for both intuitive analysis as well as further statistical or data mining analysis. The rule pruning procedure has also succeeded in eliminating all rules that are not a strong characterisation of a generated rule, implying all uninteresting results have been pruned.

5 Summary

We have addressed the problem of discovering interesting generalised association rules over categorical attributes, without assuming a taxonomy over the

attribute values. The use of a taxonomy (is-a hierarchy) for generalisation of data necessarily limits the forms of generalised rules that can be extracted, so the development of a method to discover generalised rules without this requirement has a potentially larger space of interesting rules that can be searched.

In order to do this we have integrated the rule discovery process with a form of bottom up hierarchical clustering which is designed to heuristically maximise the interestingness of the rules discovered. A concept of "interestingness" for associations was formulated which incorporates the support of a rule with the degree to which the consequent "discriminates" the antecedent from the rest of the domain.

Through applying the algorithm to an artificially generated data set, we observed that the algorithm succeeds in discovering a small number of interesting associations which are representative of associations generated in the data.

We believe this could provide a useful tool for a domain expert to gain insight into their data.

Future Work

Further development of the work should address the limitation of a single pair of attributes and the complexity of the algorithm. In order to scale the algorithm to more attributes, two methods could be developed: a rule base storing rules from multiple pairs of attributes is maintained and potentially interesting multi-attribute rules could be identified by calculating the expected interestingness of combining two-attribute rules. Alternatively, methods of merging (or splitting) across multiple attributes, as well as pruning across multiple attributes could be explored as well as identifying which attributes should form the left and right hand sides of the rules.

In its current state, a number of directions could be adopted to improve the efficiency of the algorithm. The size of the set of clusters being merged could be limited and as existing members are merged, new items are added to form new clusters. Combining a form of pruning on RHS' where uninteresting items are removed from C_i -list permanently after some point would also increase efficiency. Other clustering techniques such as establishing a hierarchical structure to allow efficient comparison of items (as adopted in BIRCH [15]) may also be adapted to the algorithm.

Acknowledgments

I thank Anna Andrusiewicz for her part in helping establish the initial direction of the work. I also thank David Truffet for his insightful comments on the algorithm.

References

1. Rakesh Agrawal, Tomasz Imielinski, and Arun Swami. Mining association rules between sets of items in large databases. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, pages 207–216, Washington, May 1993.

2. Rakesh Agrawal, Heikki Mannila, Ramakrishnan Srikant, Hannu Toivonen, and A. Inkeri Verkamo. *Advances in knowledge discovery and data mining*, chapter Fast discovery of association rules. AAAI Press/The MIT Press, 1996.
3. Anna Andrusiewicz and M. E. Orlowska. On data granularity factors that affect data mining. In *Proceedings of the 8th International Database Workshop*, 1997.
4. Ming-Sayn Chen, Jiawei Han, and Philip S. Yu. Data mining: An overview from database perspective. *IEEE Transactions on knowledge and data engineering*, 1997.
5. Martin Easter, Hans-Peter Kriegel, Jorg Sander, and Xiaowei Xu. A density based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, 1996.
6. Douglas Fisher. Improving inference through conceptual clustering. In *Proc. 1987 AAAI Conf.*, pages 461–465, July 1987.
7. Takeshi Fukuda, Yasuhiko Morimoto, Shinichi Morishita, and Takeshi Tokuyama. Data mining using two-dimensional optimized association rules: Scheme, algorithms, and visualization. In *Proceedings of the ACM-SIGMOD Conference on Management of data*, Montreal, Canada, 1996.
8. Brett Gray and M. E. Orlowska. The use of clustering to mine interesting association rules. Technical Report TR425, School of Computer Science and Electrical Engineering, The University of Queensland, 1998.
9. Jiawei Han and Yongjian Fu. Discovery of multiple-level association rules from large databases. In *Proceedings of the 21st VLDB Conference*, Zurich, Switzerland, 1995.
10. Ryszard S. Michalski and Robert E. Stepp. Automated construction of classifications: Conceptual clustering versus numerical taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5(4), July 1983.
11. Raymond T. Ng and Jiawei Han. Efficient and effective clustering methods for spacial data mining. In *Proceeding of the 20th VLDB Conference*, Santiago, Chile, 1994.
12. Ramakrishnan Srikant and Rakesh Agrawal. Mining generalized association rules. In *Proceedings of the 21st VLDB Conference*, Zurich, Swizerland, 1995.
13. Ramakrishnan Srikant and Rakesh Agrawal. Mining quantitative association rules in large relational tables. In *Proceedings of the ACM-SIGMOD Conference on Management of data*, Montreal, Canada, 1996.
14. Hannu Toivonen. Sampling large databases for association rules. In *Proceedings of the 22nd VLDB Conference*, Mumbai(Bombay), India, 1996.
15. Tian Zhang, Raghu Ramakrishnan, and Miron Livny. BIRCH: An efficient data clustering method for very large databases. In *Proceedings of the ACM-SIGMOD Conference on Management of data*, Montreal, Canada, 1996.

Selective Materialization: An Efficient Method for Spatial Data Cube Construction

Jiawei Han, Nebojsa Stefanovic, and Krzysztof Koperski

School of Computing Science
Simon Fraser University
Burnaby, BC, Canada V5A 1S6
E-mail: {han, nstefano, koperski}@cs.sfu.ca

Abstract. On-line analytical processing (OLAP) has gained its popularity in database industry. With a huge amount of data stored in spatial databases and the introduction of spatial components to many relational or object-relational databases, it is important to study the methods for spatial data warehousing and on-line analytical processing of spatial data. In this paper, we study methods for spatial OLAP, by integration of nonspatial on-line analytical processing (OLAP) methods with spatial database implementation techniques. A spatial data warehouse model, which consists of both spatial and nonspatial dimensions and measures, is proposed. Methods for computation of spatial data cubes and analytical processing on such spatial data cubes are studied, with several strategies proposed, including approximation and partial materialization of the spatial objects resulted from spatial OLAP operations. Some techniques for selective materialization of the spatial computation results are worked out, and the performance study has demonstrated the effectiveness of these techniques.

Keywords: Data warehouse, data mining, on-line analytical processing (OLAP), spatial databases, spatial data analysis, spatial OLAP.

1 Introduction

With the popular use of satellite telemetry systems, remote sensing systems, medical imaging, and other computerized data collection tools, a huge amount of spatial data has been stored in spatial databases, geographic information systems, spatial components of many relational or object-relational databases, and other spatial information repositories. It is an imminent task to develop efficient methods for the analysis and understanding of such huge amount of spatial data and utilize them effectively.

Following the trend of the development of data warehousing and data mining techniques [2, 4, 9], we propose to construct *spatial data warehouses* to facilitate on-line spatial data analysis and spatial data mining [3, 7, 11]. Similar to nonspatial data warehouses [2, 9], we consider that a spatial data warehouse is a *subject-oriented, integrated, time-variant, and non-volatile* collection of both spatial and nonspatial data in support of management's decision-making process.

In this paper, we study how to construct such spatial data warehouse and how to implement efficiently *on-line analytical processing of spatial data* (i.e., spatial

OLAP) in such warehouse environment. To motivate our study of spatial data warehousing and spatial OLAP operations, we examine the following examples.

Example 1 Regional weather pattern analysis. In British Columbia, there are over 3,000 weather probes recording temperature and precipitation for a designated small area. A user may like to view weather patterns on a map by month, by region, and by different combinations of temperature and precipitation, or even may like to dynamically drill-down or roll-up along any dimension to explore desired patterns, such as wet and hot regions in Fraser Valley in July, 1997.

Example 2 Overlay of multiple thematic maps. There often exist multiple thematic maps in a spatial database, such as altitude map, population map, and daily temperature maps of a region. By overlaying multiple thematic maps, one may find some interesting relationships among altitude, population density and temperature. For example, *flat low land in B.C. close to coast is characterized by mild climate and dense population*. One may like to perform data analysis on any selected dimension, such as drill down along a region to find the relationships between altitude and temperature.

The above examples show not only some interesting applications of spatial data warehouses but also indicate that there are many challenging issues in the implementation of spatial data warehouses. The first challenge is the construction of spatial data warehouses by integration of spatial data from heterogeneous sources and systems. Spatial data are usually stored in different industry firms and government agencies using different data formats. Data formats are not only structure-specific (e.g., raster- vs. vector-based spatial data, object-oriented vs. relational models, etc.), but also vendor-specific (e.g., ESRI, MapInfo, Intergraph, etc.). However, in this paper, we are not going to address data integration issues and we assume that a spatial data warehouse can be constructed either from a homogeneous spatial database or by integration of a collection of heterogeneous spatial databases.

The second challenge is the realization of fast and flexible on-line analytical processing in a spatial data warehouse, and this is the goal of our study. In spatial database research, spatial indexing and accessing methods have been studied extensively for efficient storage and access of spatial data [6]. Unfortunately, these methods alone cannot provide sufficient support for on-line analytical processing of spatial data because spatial OLAP operations usually summarize and characterize a large set of spatial objects in different dimensions and at different levels of abstraction, which requires fast and flexible presentation of collective, aggregated, or general properties of spatial objects. New models and techniques should be developed for on-line analysis of voluminous spatial data.

In this paper, we propose the construction of spatial data warehouse using a *spatial data cube* model (also called a *spatial multidimensional database* model). A *star/snowflake model* [2] is used to build a spatial data cube which consists of some spatial dimensions and/or measures together with nonspatial ones. Methods for efficient implementation of spatial data cubes are examined with some interesting techniques proposed, especially on precomputation and

selective materialization of spatial OLAP results. The precomputation of spatial OLAP results, such as merge of a number of spatially connected regions, is important not only for fast response in result display but also for further spatial analysis and spatial data mining [3, 7, 11].

2 A model of spatial data warehouses

Unlike relational or entity-relationship models which are used for designing databases for ad-hoc querying and on-line transaction processing, data warehouse is designed for on-line analytical processing and business decision making, and it usually adopts a star schema model [2, 10], where the data warehouse contains a large central table (*fact table*) and a set of smaller attendant tables (*dimensional tables*) displayed in a radial pattern around the central table. The fact table stores the keys of multiple dimensions and the numerical measures of the business. The dimensional tables are where the textual description of the dimensions of the business are stored. A variant of a star schema model is called a snowflake (schema) model [2, 10], where some dimension tables are normalized, further splitting into more tables, forming the shape similar to a piece of snowflake. With such a star/snowflake schema model, multi-dimensional databases or data cubes [1, 8] can be constructed to facilitate typical OLAP operations such as *drill-down*, *roll-up*, *dicing*, *slicing*, *pivoting*, etc.

To model spatial data warehouses, the star/snowflake schema model is still considered to be a good choice because it provides a concise and organized warehouse structure and facilitates OLAP operations. In a spatial warehouse, both dimensions and measures may contain spatial components. A spatial data cube can be constructed according to the dimensions and measures modeled in the warehouse.

There are three cases for modeling *dimensions* in a spatial data cube:

1. Nonspatial dimension is a dimension containing only nonspatial data. For example, two dimensions, *temperature* and *precipitation*, can be constructed for the warehouse in Example 1, each is a dimension containing nonspatial data whose generalizations are nonspatial, such as *hot*, and *wet*.
2. Spatial-to-nonspatial dimension is a dimension whose primitive level data is spatial but whose generalization, starting at a certain high level, becomes nonspatial. For example, *state* in the U.S. map is represented by spatial data. However, each state can be generalized to some nonspatial value, such as *pacific_northwest*, or *big_state*, and its further generalization is nonspatial, and thus playing a similar role as a nonspatial dimension.
3. Spatial-to-spatial dimension is a dimension whose primitive level and all of its high-level generalized data are spatial. For example, *equi-temperature-region* in Example 2 is represented by spatial data, and all of its generalized data, such as regions covering *0-5_degree*, *5-10_degree*, and so on, are also spatial.

Notice that the last two cases indicate that a spatial attribute, such as *county*, may have more than one way to be generalized to high level concepts, and the generalized concepts can be spatial, such as *map representing larger regions*, or nonspatial, such as *area* or *general description of the region*.

In addition, a computed measure can be used as a dimension in a warehouse, which we call a *measure-folded dimension*. For example, the measure *monthly average temperature in a region* can be treated as a dimension and can be further generalized to a value range or a descriptive value, such as *cold*. Moreover, a dimension can be specified by experts/users based on the relationships among attributes or among particular data values, or be generated automatically based on spatial data analysis techniques [3, 7, 11].

We distinguish two cases for modeling *measures* in a spatial data cube.

1. **Numerical measure** is a measure containing only numerical data. For example, one measure in a spatial data warehouse could be *monthly revenue* of a region, and a roll-up may get the total revenue by year, by county, etc. Numerical measures can be further classified into *distributive*, *algebraic*, and *holistic* [5]. The scope of our discussion related to numerical measures is confined to *distributive* and *algebraic* measures.
2. **Spatial measure** is a measure which contains one or a collection of pointers to spatial objects. For example, during the generalization (or roll-up) in a spatial data cube of Example 1, the regions with the same range of *temperature* and *precipitation* will be grouped into the same cell, and the measure so formed contains a collection of pointers to those regions.

A nonspatial data cube contains only nonspatial dimensions and numerical measures. If a spatial data cube contained spatial dimensions but no spatial measures, its OLAP operations, such as drilling or pivoting, could be implemented in a way similar to nonspatial data cubes. However, the introduction of spatial measures to spatial data cubes raises some challenging issues on efficient implementation, which will be the focus of this study.

The star model of Example 1 and its corresponding dimensions and measures are illustrated as follows.

Example 3. A star model can be constructed, as shown in Figure 1, for the *BC_weather* warehouse of Example 1, where the B.C. map with regions covered by many weather probes is shown too. The warehouse consists of four dimensions: *temperature*, *precipitation*, *time*, and *region_name*, and three measures: *region_map*, *area*, and *count*.

A concept hierarchy for each dimension can be created by users or experts or generated automatically by data clustering or data analysis. An example of concept hierarchy for *temperature* dimension is shown in Figure 2. Of the three measures, *region_map* is a *spatial* measure which represents a collection of spatial pointers pointing to the corresponding regions, *area* is a *numerical* measure which represents the sum of the total areas of the corresponding spatial objects, and *count* is a *numerical* measure which represents the total number of base regions (probes) accumulated in the corresponding cell.

Table 1 shows a data set that may be collected from a number of weather probes scattered in B.C. For example, *region_name AM08* may represent an area of *Burnaby mountain*, and whose generalization could be *North_Burnaby*, and then *Burnaby*, *Greater_Vancouver*, *Lower_Mainland*, and *Province_of_BC*, each corresponding to a region on the B.C. map.

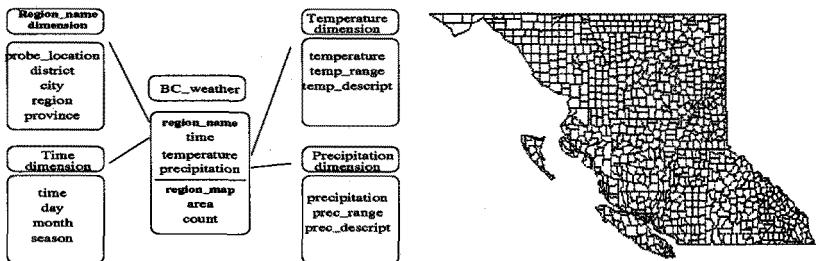


Fig. 1. Star model of a spatial data warehouse: BC_weather

Temperature:

- any ⊃ (cold, mild, hot)
- cold ⊃ (below -20, -20 to -10, -10 to 0)
- mild ⊃ (0 to 10, 10 to 15, 15 to 20)
- hot ⊃ (20 to 25, 25 to 30, 30 to 35, above 35)

Fig. 2. Hierarchy for *temperature* dimension in BC_weather

Let us examine some popular OLAP operations and analyze how they are performed in a spatial data cube.

1. *Slicing and dicing*, which of each selects a portion of the cube based on the constant(s) in one or a few dimensions. This can be realized by transforming the selection criteria into a query against the spatial data warehouse and be processed by query processing methods [6].
2. *Pivoting*, which presents the measures in different cross-tabular layouts. This can be implemented in a similar way as in nonspatial data cubes.
3. *Roll-up*, which generalizes one or a few dimensions (including the removal of some dimensions when desired) and performs appropriate aggregations in the corresponding measure(s). For nonspatial measures, aggregation is implemented in the same way as in nonspatial data cubes [1, 5, 13]. However, for spatial measures, aggregation takes a collection of a spatial pointers in a map or map-overlay and performs certain *spatial aggregation* operation, such as region merge, or map overlay. It is challenging to efficiently implement

Region_name	Time	Temperature	Precipitation
AA00	01/01/97	-4	1.5
AA01	01/01/97	-7	1.0
...
AA00	01/02/97	-6	2.5
AA01	01/02/97	-8	1.0
...

Table 1. Table for weather probes

such operations since it is both time and space consuming to compute spatial merge or overlay and save the merged or overlaid spatial objects. This will be discussed in detail in later sections.

4. *Drill-down*, which specializes one or a few dimensions and presents low-level objects, collections, or aggregations. This can be viewed as a reverse operation of *roll-up* and can often be implemented by saving a low level cube, presenting it, or performing appropriate generalization from it when necessary.

From this analysis, one can see that a major performance challenge for implementing spatial OLAP is the efficient construction of spatial data cubes and implementation of roll-up/drill-down operations. This can be illustrated by analysis of how OLAP is performed in the data cube of Example 3.

Example 4. The roll-up of the data cube of B.C. weather probe of Example 1 can be performed as follows.

The roll-up on the *time* dimension is performed to roll-up values from day to month. Since temperature is a *measure-folded* dimension, the roll-up on the *temperature* dimension is performed by first computing the *average temperature grouped by month and by spatial region* and then generalizing the values to ranges (such as -10 to 0) or to descriptive names (such as "cold"). Similarly, one may roll-up along the precipitation dimension by computing the average precipitation grouped by month and by region. The *region_name* dimension can be dropped if one does not want to generalize data according to specified regions. By doing so, the generalized data cube contains three dimensions, *time (in month)*, *temperature (in monthly average)*, and *precipitation (in monthly average)*, and one spatial measure which is a collection of spatial object_ids, as shown in Table 2.

Time	Temperature	Precipitation	Collection of Spatial_ids
March	cold	0.1 to 0.3	{AL04, AM03, ... XN87}
March	cold	0.3 to 1.0	{AM10, AN05, ... YP90}
...

Table 2. Result of a roll-up operation

Different roll-ups from the B.C. weather map data (Figure 1) produce two different generalized region maps as shown in Figure 3, each being the result of merging a large number of small (probe) regions. Computing such spatial merges of a large number of regions flexibly and dynamically poses a major challenge to the implementation of spatial OLAP operations. Only if appropriate precomputation is performed, can the response time be satisfactory to users.

Similar to the structure of a nonspatial data cube [2, 13], a spatial data cube consists of a lattice of cuboids, with the lowest one (*base cuboid*) references all the dimensions at the primitive abstraction level (i.e., group-by all the dimensions), and the highest one (*apex point*) summarizes all the dimensions at the top-most

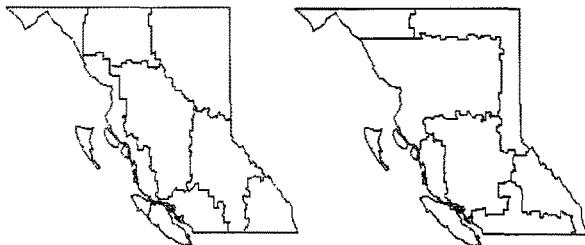


Fig. 3. Generalized regions after different roll-up operations

abstraction level (i.e., no group-by's in aggregation). Drill-down, roll-up, and dimension reduction in a spatial data cube result in different cuboids in which each cell contains the aggregation of measure values or clustering of spatial object pointers. The aggregation (such as sum, average, etc.) of numeric values results in a new numeric value. However, the clustering of spatial object pointers may not lead to a single, new spatial object. If all the objects pointed to by the spatial pointers are connected, they can be merged into one large region; otherwise, they will be represented by a set of regions. A numeric value usually takes only about two- to eight-byte storage space and a small amount of time in computation. However, a spatial object may take kilo- to mega-bytes in storage space and it is much more costly to compute the merge or overlay of a set of spatial regions than its numerical counterpart.

Furthermore, one may expect that OLAP operations, such as drilling or pivoting, be performed flexibly in a spatial data warehouse with fast response time since a user may like to interact with the system to obtain necessary statistics for decision making. Instead of computing such aggregations on-the-fly, it is often necessary to precompute some high-level views (*cuboids* [1]) and save them in the database as *materialized views* (*computed cuboids*) to facilitate fast OLAP operations.

There are different products in (*nonspatial*) data warehouse industry: some materialize every cuboid, some none, and some only part of the cube (i.e., some of the cuboids). There are interesting studies on efficient computation of data cubes [1, 13]. A previous study [8] shows that materializing every view requires a huge amount of disk space, whereas not materializing any view requires a great deal of on-the-fly, and often redundant, computation. The study promotes partial materialization as a solution and an algorithm has been proposed to determine what should be precomputed for partial materialization of data cubes.

In the implementation of spatial data cubes, we face a dilemma of balancing the cost of on-line computation and the storage overhead of storing computed spatial measures: the substantial computation cost for on-the-fly computation of spatial aggregations calls for precomputation but substantial overhead for storing aggregated spatial values discourages it. Obviously, we should not materialize every cuboid with limited storage space but we cannot afford to compute all the spatial aggregates on-the-fly.

This motivates us to propose interesting techniques for selective materialization of spatial data cubes. In the previous OLAP studies, granularity of data cube materialization has been at the cuboid level, that is, either completely materialize a cuboid or not at all. However, for materialization of spatial measure, it is often necessary to consider finer granularity and examine individual cells to see whether a group of spatial objects within a cell should be precomputed. We examine this technique in detail in the next section.

3 Methods for Computing Spatial Measures in Spatial Data Cube Construction

In this discussion, we assume that the computation of spatial measures involves *spatial region merge* operation only. The principles discussed here, however, are also applicable to other kinds of spatial operations, such as *spatial map overlay*, *spatial join* [6], and *intersection* between lines and regions.

3.1 Choices for computation of spatial measures

There are at least three possible choices regarding the computation of spatial measures in spatial data cube construction:

1. Collect and store the corresponding spatial object pointers but do not perform precomputation of spatial measures in a spatial data cube.

This can be implemented easily by storing in the corresponding cube cell a pointer to a collection of spatial object pointers. This choice indicates that the (region) merge of a group of spatial objects, when necessary, may have to be performed on-the-fly. If the OLAP results are used only for viewing, display-only mode could be useful. However, they can be used for further spatial analysis and spatial data mining, such as spatial association, classification, etc. [7, 11] and it is important to merge a number of spatially connected regions for such analysis.

2. Precompute and store some rough approximation/estimation of the spatial measures in a spatial data cube.

This choice is good for a rough view or coarse estimation of spatial merge results under the assumption that it takes little storage space to store the coarse estimation result in the worst case. For example, the minimum bounding rectangle (MBR) of the spatial merge result can be taken as a rough estimate of a merged region. If higher precision is needed for specific cells, the system can either fetch precomputed high quality results, if available, or compute them on-the-fly.

3. Selectively precompute some spatial measures in a spatial data cube.

This seems to be a smart choice, but the question is how to select a set of spatial measures for precomputation. The selection can be performed at the cuboid level, i.e., either precompute and store *every* set of mergeable spatial regions for *every* cell of a selected cuboid, or precompute none if the cuboid is not selected. Since a cuboid usually covers the whole map, it may involve precomputation and storage of a large number of mergeable spatial

objects but some of them could be rarely used. Therefore, it is recommended to perform selection at a finer granularity level by examining each group of mergeable spatial objects in a cuboid to determine whether such a merge should be precomputed.

Based on the above analysis, our discussion is focused on how to select a group of mergeable spatial objects for precomputation from a set of targeted spatial data cuboids.

3.2 Methods for selection of mergeable spatial objects for materialization

We now examine the data cube structure in more detail. Let the data cube consist of n dimensions, D_1, \dots, D_n , where the i -th dimension D_i has k_i levels of hierarchy, and the top level has only one special node “any” which corresponds to the removal of the dimension. For an n -dimensional data cube, if we allow new cuboids to be generated by climbing up the hierarchies along each dimension the total number of cuboids that can be generated is, $N = \prod_{i=1}^n k_i - 1$. This is a big number, thus it is recommended to materialize only *some* of the possible cuboids that can be generated.

Three factors may need to be considered when judging whether a cuboid should be selected for materialization: (1) the potential access frequency of the generated cuboid, (2) the size of the generated cuboid, and (3) how the materialization of one cuboid may benefit the computation of other cuboids in the lattice [8]. A greedy cuboid-selection algorithm has been presented in [8] based on the analysis of the latter two factors. A minor extension to the algorithm may take into consideration the first factor, the potential access frequency of the generated cuboid, where the potential access frequency can be estimated by an expert or a user, or calculated based on the cube access history.

The analysis of whether a cuboid should be selected for precomputation in a spatial data cube is similar to a nonspatial one although an additional factor, the cost of on-line computation of a particular spatial cuboid, should be considered in the cost estimation since the spatial computation, such as region merging, map overlaying, spatial join, could be expensive when involving a large number of spatial objects. Even when we decide to materialize a cuboid, it is still unrealistic to compute and store every spatial measure for each cell because it may consume a substantial amount of computation time and disk space, especially considering many of them may not be examined in any detail or may only be examined a small number of times.

In our subsequent analysis, we assume that a set of cuboids have been selected for materialization using an extended cuboid-selection algorithm similar to the one in [8] and examine how to determine which sets of mergeable spatial objects should be precomputed. Two algorithms, *pointer intersection* and *object connection*, are worked out for selection of precomputed objects. The general idea of the algorithms is as follows. Given a set of selected cuboids each associated with an (estimated) access frequency, and *min_freq* (the minimum access frequency threshold), a set of mergeable objects should be precomputed if and only if its

expected access frequency is no less than *min_freq*. Notice that a merged object also counts as one to be accessed if it is used to construct a larger object which is not precomputed. The difference between the two algorithms is that the former (*pointer intersection*) first computes the intersections among the sets of object pointers, and then performs threshold filtering and examines their corresponding spatial object connections; whereas the latter (*object connection*) examines the corresponding spatial object connections before threshold filtering.

Algorithm 3.1 (Pointer Intersection) A pointer intersection method for the selection of a group of candidate connected regions for precomputation and storage of the merge results during the construction of a spatial data cube.

- Input:**
- A cube lattice which consists of a set of selected cuboids obtained by running an extended cuboid-selection algorithm similar to [8].
 - An *access frequency table* which registers the access frequency of each cuboid in the lattice.
 - A group of spatial pointers in each cell of a cuboid in the lattice.
 - A region map which illustrates the neighborhood of the regions. The information is collected in *obj_neighbor* table in the format of (*object_pointer*, *a_list_of_neighbors*).
 - *min_freq*: A threshold which represents the minimum accumulated access frequency of a group of connected regions (in order to be selected for precomputation).

Output: A set of candidate groups selected for spatial precomputation.

Method:

- The main program is outlined as follows, where *max_cuboid* is the maximum number of cuboids selected for materialization.
- ```

(1) FOR cuboid_i := 1 TO max_cuboid DO
(2) FOR cuboid_j := cuboid_i TO max_cuboid DO
(3) FOR EACH cell_i IN cuboid_i DO
(4) get_max_intersection(cell_i, cuboid_j, candidate_table);
(5) frequency_computing_&_filtering(candidate_table);
(6) spatial_connectivity_testing(candidate_table, connected_obj_table);
(7) shared_spatial_merging(connected_obj_table, merged_obj_table);

```
- The procedure *get\_max\_intersection*(*cell\_i*, *cuboid\_j*, *candidate\_table*) is to calculate intersections of maximal cardinality between a cell (*cell\_i*) from one cuboid with all cells from another cuboid (*cuboid\_j*) and insert them into the candidate table.

For more detailed explanation of the algorithm steps and several optimization techniques used we refer to [12].

**Rationale of the algorithm.** The algorithm is to find those spatial object groups in the data cube which are frequently accessed and mergeable, and then perform spatial merge for them in precomputation. The algorithm works on the candidate cuboids that are selected based on an extension to a well-known cuboid selection algorithm [8]. Lines (1) to (4) ensure that every pair of cuboids is examined for

each candidate cube cell which derives the maximal intersections of spatial object pointers and stores them into candidate table. Note that a self-intersection for cuboids is performed as well. Line (5) removes from the candidate table those candidates whose accumulated access frequency is less than *min\_freq*. Line (6) finds the spatially connected subsets from each candidates object pointer set and line (7) materializes them and puts them into the *merged\_obj\_table*.

**Note.** We now clarify the reason for applying the self-intersection in this algorithm (Lines (1) and (2)), since it might not be obvious to a reader. There can be a number of groups that appear in a single, yet frequent cuboid and it is important that such groups be identified. Were a self-intersection not applied, these groups would be skipped. The performance analysis, conducted in Section 4, will show that the self-intersection has a significant positive impact on the effectiveness of the algorithm (see Figure 4).

**Algorithm 3.2 (Object Connection)** A object connection method for the selection of a group of candidate connected regions for precomputation and storage of the merge results during the construction of a spatial data cube.

**Input:** The same as Algorithm 3.1.

**Output:** The same as Algorithm 3.1.

**Method:**

- The main program is different from that of Algorithm 3.1 at Line (4) where connection is immediately checked before proceeding further. Thus, the old Line (6) is removed since it has been done in Line (4).

```
(1) FOR cuboid_i := 1 TO max_cuboid 1 DO
(2) FOR cuboid_j := cuboid_i TO max_cuboid DO
(3) FOR EACH cell_i IN cuboid_i DO
(4) get_max_connected_intersection(cell_i, cuboid_j,
 candidate_connected_obj_table);
(5) frequency_computing_&_filtering(candidate_connected_obj_table);
(6) shared_spatial_merging(candidate_connected_obj_table,
 merged_obj_table);
```

**Rationale of the algorithm.** The major difference of this algorithm from the former one, is at the handling of connected objects. The first algorithm delays the connectivity checking after *min\_freq* threshold filtering, whereas the new one does it at the insertion into the candidate table. By looking at the algorithms, one may think that they produce identical results in terms of selected regions for premerge. In the subsequent discussion we will show that it may not always be the case.

Suppose that groups *A* and *B* are detected by *get\_max\_intersection* procedure in the *pointer intersection* algorithm. Moreover, let us assume that both groups contain a common connected subgroup *C*. Since groups are checked for spatial connectivity only after the frequency filtering step is applied this algorithm may not detect group *C*. *Object connection* algorithm will detect group *C* in its *get\_max\_connected\_intersection* procedure and since its frequency may be

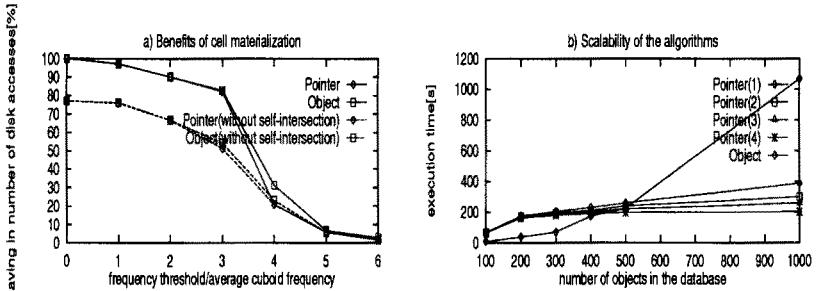
higher than that for groups *A* and *B* and it may be merged. We provide a more formal explanation in a form of a theorem [12].

## 4 Performance Analysis

In the previous section, we presented two algorithms for selective materialization of spatial measures: *pointer intersection* algorithm, and *object connection* algorithm. In order to evaluate and compare their performances, we implemented the algorithms and conducted a simulation study.

We first analyze *effectiveness* of the algorithms. Since the goal of selective materialization is to enhance on-line processing, we study the usability of materialized groups. The two algorithms are compared with respect to their effectiveness in Figure 4a). Here, we analyze materialization benefit as a function of *frequency threshold/average cuboid frequency* ratio. The figure reveals following:

- The benefits of both algorithms decrease with the increase of frequency threshold.
- There is only a slight difference between effectiveness of the two algorithms.
- If self-intersection is not applied, the benefits do not converge to 100%.



**Fig. 4.** Performance comparison for the algorithms

It was expected that a higher frequency threshold leads to the smaller number of premerged objects and thus to the smaller benefit. When frequency threshold is not applied benefits for both algorithms reach 100%. The difference between effectiveness of the algorithms is more or less marginal in our simulation, however only the real world application with a large number of spatial objects can confirm a potential trend. As we explained in our presentation of *pointer intersection* and *object connection* algorithms, performing the self-intersection on cuboids vastly improves the benefits of materialization. If the self-intersection were not applied, at most 77% of disk accesses would be avoided during on-line processing. When the frequency threshold increases the differences become marginal.

Figure 4b) shows the efficiency comparison of precomputation running time for the algorithms. Although precomputation running time is not even distantly as crucial as on-line running time, we are still concerned with precomputation efficiency mainly due to the need for warehouse maintenance. The number of cuboids was set to 10 for this experiment. We varied (1) the number of objects

(from 100 to 1000), and (2) *frequency threshold/average cuboid frequency* ratio (from 1 to 4). When the number of objects is small, *object connection* algorithm has an edge over *pointer intersection* algorithm. However, by increasing the number of objects, the performance of *object connection* algorithm significantly deteriorates, while *pointer intersection* algorithm shows little sensitivity to the number of objects.

Another observation from Figure 4b) is that the frequency threshold is irrelevant for the execution time of *object connection* algorithm. This was expected since frequency filtering is the last step in the algorithm. On the contrary, *pointer intersection* algorithm shows slightly better performance when the frequency threshold increases due to fewer groups tested for spatial connectivity.

We also compared the two algorithms with respect to scalability to the number of cuboids in the spatial data cube and the experiments showed similar results [12].

We anticipate that future spatial data warehouses will be built on the following two premises: large number of objects and relatively small number of frequent queries. According to the conducted experiments, we believe that *pointer intersection* algorithm fits better than *object connection* algorithm into such data warehouse environment. Typical applications that confirm to above assumptions are regional weather pattern analysis, demographic analysis, real estate business, etc. However, if a spatial data warehouse is to contain few objects a data warehouse designer could choose applying *object connection* algorithm for selective materialization of spatial measures.

## 5 Discussion

We have proposed a *spatial data cube*-based data warehouse model to facilitate efficient OLAP operations on spatial data. A spatial data cube consists of both spatial and nonspatial dimensions and measures. Since it is challenging to compute spatial measures in such a data cube, our study has been focused on the method for selective materialization of spatial measures. In comparison with previous studies of selective materialization of data cubes, such as [8], a major difference of our approach is the granularity of selective materialization. Although the uniform materialization of numeric measures in a nonspatial cuboid takes a reasonable amount of space and computational overhead, uniform materialization of spatial measures in a spatial data cube may lead to an explosive growth of both storage and computation costs, even for a selected subset of cuboids.

In this study, a method is proposed to perform selective materialization based on the relative access frequency of the sets of mergeable spatial regions, that is, the sets of mergeable spatial regions should be precomputed if they are expected to be accessed/used frequently. Here we discuss some issues related to the proposed method and some possible alternatives.

We assume the existence of some information about the access frequencies of a set of selected cuboids. What if there exists no such information initially? One solution is to assign an initial access frequency only to a *level* in the lattice (less work than assigning it to every *cuboid*), based on some data semantics or simply

assuming that medium levels (such as *county* level in a province map) are accessed most frequently. Such a frequency estimate can be adjusted based on later accessing records. Alternatively, one may choose to materialize every mergeable group accessed initially, record their access frequencies, and later throw away the rarely used groups. We have assumed the uniform access frequency for all cells of each cuboid. This may not be true in real cases. For example, *Southern\_B.C.* is often accessed more frequently than *Northern\_B.C.* due to that the population and business are clustered in the former. In such cases, the granularity of access frequency assignment can be tuned to a subset of a cuboid, with little change to the method itself.

Our method selects groups of regions for precomputation based on the access frequency but not on the concrete size of the mergeable regions. This is reasonable if all the base regions are of comparable size and complexity. However, in some databases, some regions could be substantially larger or more complicated and thus take substantially larger space and more computation than others. Taking this factor into consideration, the judgment criteria can be modified accordingly, for example, adding the cost of region merging and a compression ratio which is the difference in size between the precomputed vs. not precomputed regions.

In our analysis, we have adopted the philosophy and algorithm of selective materialization of data cubes as proposed in [8]. Is it possible to use our analysis to the approaches which attempt to materialize every cuboid in a data cube [1]? Yes, it is reasonable, because even if it is sometimes plausible to materialize every cuboid for nonspatial measures, it is rarely possible to materialize every cell of the cuboids for spatial measures since each cell contains groups of large spatial objects. Therefore, selective materialization of mergeable spatial objects for OLAP is essential for implementation of spatial OLAP operations.

Selective materialization is a trade-off between on-line OLAP computation time and the overall storage space. However, even if we always make a good selection, there are still groups whose spatial aggregates are not precomputed. Spatial merge, join, or overlay are expensive operations and their on-line computation may slow down the system substantially. A possible compromise is to adopt a multi-resolution approach which distinguishes high and low resolution computations.

## 6 Conclusions

We have studied the construction of spatial data warehouses based on a spatial data cube model which consists of both spatial and nonspatial dimensions and measures. For OLAP on spatial data, we studied issues on efficient computation of spatial measures and proposed methods for object-based selective materialization of spatial OLAP computation results.

Furthermore, we feel that the principles of object-based selective materialization for computation of data cubes are not confined to spatial data only. Other databases which handle complex objects, such as multimedia databases, engineering design databases, will encounter similar problems and it is essential

to perform object-based selective materialization as a space/time trade-off for efficient OLAP operations.

With the recent success and great promise of OLAP technology, spatial OLAP holds a high promise for fast and efficient analysis of large amount of spatial data. Thus, spatial OLAP is expected to be a promising direction for both research and development in the years to come. The development of new spatial database technology for spatial OLAP, the object-based selective materialization of other spatial measures, including spatial overlay and spatial joins, the efficient storage and indexing of partially materialized spatial data cubes, and the smooth integration of spatial OLAP with spatial data mining are interesting issues for future research.

## References

1. S. Agarwal, R. Agrawal, P. M. Deshpande, A. Gupta, J. F. Naughton, R. Ramakrishnan, and S. Sarawagi. On the computation of multidimensional aggregates. In *Proc. 1996 Int. Conf. Very Large Data Bases*, pages 506–521, Bombay, India, Sept. 1996.
2. S. Chaudhuri and U. Dayal. An overview of data warehousing and OLAP technology. *ACM SIGMOD Record*, 26:65–74, 1997.
3. M. Ester, H.-P. Kriegel, and J. Sander. Spatial data mining: A database approach. In *Proc. Int. Symp. Large Spatial Databases (SSD'97)*, pages 47–66, Berlin, Germany, July 1997.
4. U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy. *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press, 1996.
5. J. Gray, S. Chaudhuri, A. Bosworth, A. Layman, D. Reichart, M. Venkatrao, F. Pellow, and H. Pirahesh. Data cube: A relational aggregation operator generalizing group-by, cross-tab and sub-totals. *Data Mining and Knowledge Discovery*, 1:29–54, 1997.
6. R. H. Güting. An introduction to spatial database systems. *The VLDB Journal*, 3:357–400, 1994.
7. J. Han, K. Koperski, and N. Stefanovic. GeoMiner: A system prototype for spatial data mining. In *Proc. 1997 ACM-SIGMOD Int. Conf. Management of Data*, pages 553–556, Tucson, Arizona, May 1997.
8. V. Harinarayan, A. Rajaraman, and J. D. Ullman. Implementing data cubes efficiently. In *Proc. 1996 ACM-SIGMOD Int. Conf. Management of Data*, pages 205–216, Montreal, Canada, June 1996.
9. W. H. Inmon. *Building the Data Warehouse*. John Wiley, 1996.
10. R. Kimball. *The Data Warehouse Toolkit*. John Wiley & Sons, New York, 1996.
11. K. Koperski, J. Han, and J. Adhikary. Mining knowledge in geographic data. In *Comm. ACM (to appear)*, 1998.
12. N. Stefanovic. Design and implementation of on-line analytical processing (OLAP) of spatial data. *M.Sc. Thesis*, Simon Fraser University, Canada, September 1997.
13. Y. Zhao, P. M. Deshpande, and J. F. Naughton. An array-based algorithm for simultaneous multidimensional aggregates. In *Proc. 1997 ACM-SIGMOD Int. Conf. Management of Data*, pages 159–170, Tucson, Arizona, May 1997.

# Mining Market Basket Data Using Share Measures and Characterized Itemsets

Robert J. Hilderman, Colin L. Carter, Howard J. Hamilton, and Nick Cercone

Department of Computer Science

University of Regina

Regina, Saskatchewan, Canada, S4S 0A2

{hilder,carter,hamilton,nick}@cs.uregina.ca

**Abstract.** We propose the *share-confidence framework* for knowledge discovery from databases which addresses the problem of mining itemsets from market basket data. Our goal is two-fold: (1) to present new itemset measures which are practical and useful alternatives to the commonly used support measure; (2) to not only discover the buying patterns of customers, but also to discover customer profiles by partitioning customers into distinct classes. We present a new algorithm for classifying itemsets based upon characteristic attributes extracted from census or lifestyle data. Our algorithm combines the *Apriori* algorithm for discovering association rules between items in large databases, and the *AOG* algorithm for attribute-oriented generalization in large databases. We suggest how characterized itemsets can be generalized according to concept hierarchies associated with the characteristic attributes. Finally, we present experimental results that demonstrate the utility of the share-confidence framework.

## 1 Introduction

Consider a retail sales operation with a large inventory consisting of many distinct products. The operation is situated in a location where the customer base is socio-economically diverse, with annual household incomes ranging from very low to very high, and demographically ranging from young families to the elderly. The sales manager has used data mining to determine those products that are typically purchased together and those that are most likely to be purchased given that particular products have already been selected (called *itemsets* [2, 14]). Analysis of the itemsets has enabled him to strategically arrange store displays and plan advertising campaigns to increase sales. He now wonders whether there are any more subtle socio-economic buying patterns that could be helpful in guiding the distribution of flyers during the next advertising campaign. For example, he would like to know which itemsets are more likely to be purchased by those with specific incomes or by those with children. He would also like to know which itemsets are more likely to be purchased by those living in particular neighborhoods. He believes that characterizing itemsets with classificatory information available from credit card or cheque transactions will allow him to answer queries of this kind.

In this paper, we propose the *share-confidence framework* that looks beyond the simple frequency with which two or more items are bought together. We introduce a new algorithm, called *CI*, which integrates the *Apriori* algorithm for discovering association rules between items in large databases [2, 1], and the *AOG* algorithm for attribute-oriented generalization in large databases [9, 11]. We also show how market basket data can be mined using share measures and characterized itemsets which have been generalized according to concept hierarchies associated with characteristic attributes. However, it should be noted that our methods are not limited to the discovery of customer profiles based upon market basket data, the method is more widely applicable to any problem where taxonomic hierarchies can be associated with characterized data.

The remainder of this paper is organized as follows. In Section 2, we present a formal description of the market basket analysis problem and introduce the share-confidence framework. In Section 3, we describe characterized itemsets and an algorithm for generating characterized itemsets from market basket data. In Section 4, we present experimental results obtained using the share-confidence framework on a database supplied by a commercial partner. We conclude in Section 5 with a summary of our work.

## 2 The Share-Confidence Framework

The problem of discovering association rules form market basket data has been formally defined as follows [2]. Let  $I = \{i_1, i_2, \dots, i_m\}$  be a set of literals, called *items*. Let  $D$  be a set of *transactions*, where each transaction  $T$  is an *itemset* such that  $T \subseteq I$ . Transaction  $T$  contains  $X$ , a set of some items in  $I$ , if  $X \subseteq T$ . An *association rule* is an implication of the form  $X \Rightarrow Y$ , where  $X \subset I$ ,  $Y \subset I$ , and  $X \cap Y = \emptyset$ . The association rule  $X \Rightarrow Y$  holds in transaction set  $D$  with *confidence*  $c$ , if  $c\%$  of transactions in  $D$  that contain  $X$ , also contain  $Y$ . The association rule  $X \Rightarrow Y$  has *support*  $s$  in transaction set  $D$ , if  $s\%$  of transactions in  $D$  contain  $X \cup Y$ . This formalism is the *support-confidence framework* [4].

The most studied and analyzed algorithm for generating itemsets in the support-confidence framework is *Apriori*, described in detail in [1, 2, 3]. This algorithm extracts the set of frequent itemsets from the set of candidate itemsets generated. A *frequent itemset* is an itemset whose support is greater than some user-specified minimum and a *candidate itemset* is an itemset whose support has yet to be determined. *Apriori* combines the frequent itemsets from pass  $k - 1$  to create the candidate itemsets in pass  $k$ . It has the important property that if any subset of a candidate itemset is not a frequent itemset, then the candidate itemset is also not a frequent itemset.

In the support-confidence framework, the purchase of an item is indicated by a binary flag (i.e., the item is either purchased or not purchased). From this binary flag, we can determine the number of transactions containing an itemset, but not the number of items in the itemset. If we know the number of items, we may find that an itemset is actually more frequent than support indicates, allowing for more accurate financial analysis, comparisons, and projections. Since

support does not consider quantity and value, its use is limited as a practical indicator for determining the financial implications of an itemset.

We will now extend the formalization of the market basket problem. The problem definition is identical to that for the support-confidence framework, except that we introduce the notion of share for itemsets, and redefine the notions of frequent itemsets and confidence. We refer to this extended formalism as the *share-confidence framework*, introduced in [8] as *share measures*.

In the sections that follow, we define the functions upon which the share-confidence framework is based. For the examples, refer to the transaction database shown in Table 1 and the item database shown in Table 2. In Table 1, the *TID* column describes the transaction identifier and columns *A* to *F* describe the items (products) being sold. Note that binary values are not used to indicate the purchase of an item, instead the actual number of items purchased in the corresponding transaction (i.e., the counts) is used. In Table 2, the *Item* column describes the valid items and the *Retail Price* column describes the retailer's selling price to the customer.

**Table 1.** An example transaction database with counts

| <i>TID</i>             | <i>A</i> | <i>B</i> | <i>C</i> | <i>D</i> | <i>E</i> | <i>F</i> |
|------------------------|----------|----------|----------|----------|----------|----------|
| <i>T</i> <sub>1</sub>  | 1        | 0        | 2        | 2        | 0        | 0        |
| <i>T</i> <sub>2</sub>  | 0        | 3        | 0        | 0        | 1        | 0        |
| <i>T</i> <sub>3</sub>  | 4        | 1        | 2        | 0        | 0        | 0        |
| <i>T</i> <sub>4</sub>  | 0        | 0        | 3        | 0        | 1        | 2        |
| <i>T</i> <sub>5</sub>  | 0        | 0        | 0        | 4        | 0        | 1        |
| <i>T</i> <sub>6</sub>  | 0        | 3        | 2        | 0        | 1        | 0        |
| <i>T</i> <sub>7</sub>  | 3        | 0        | 0        | 1        | 2        | 4        |
| <i>T</i> <sub>8</sub>  | 2        | 0        | 0        | 4        | 0        | 2        |
| <i>T</i> <sub>9</sub>  | 0        | 1        | 0        | 0        | 2        | 1        |
| <i>T</i> <sub>10</sub> | 0        | 4        | 1        | 0        | 1        | 0        |
| <i>T</i> <sub>11</sub> | 0        | 0        | 3        | 0        | 0        | 2        |
|                        | 10       | 12       | 13       | 11       | 8        | 12       |

**Table 2.** The item database

| <i>Item</i> | <i>Retail Price</i> |
|-------------|---------------------|
| <i>A</i>    | 1.50                |
| <i>B</i>    | 2.25                |
| <i>C</i>    | 5.00                |
| <i>D</i>    | 4.75                |
| <i>E</i>    | 10.00               |
| <i>F</i>    | 7.50                |

## 2.1 Preliminary Definitions

The definitions in this section were implemented in a data mining system for analyzing market basket data. This system is an extension of *DB-Discover*, a software tool for knowledge discovery from databases [7, 6]. Definitions 1 to 6 are used to query summary views containing discovered frequent itemsets.

**Definition 1.** The *local itemset count* is the sum of the local item counts (i.e., the quantity of a particular item purchased in a particular transaction) for all

transactions which contain a particular item in a particular itemset, denoted as  $lisc(i, x)$ , where  $lisc(i, x) = \sum lic(i, t_k)$ ,  $lic(i, t)$  is the value at the intersection of row  $t$  and column  $i$ ,  $i \in I$ ,  $x \subseteq I$ ,  $x \in t_k$ , and  $t_k \in D$ .

**Query.** “Give the quantity of item  $C$  in itemset  $\{B, C\}$ .”

**Result.** The local itemset count for item  $C$  in itemset  $\{B, C\}$  is  $lisc(C, \{B, C\}) = lic(C, T_3) + lic(C, T_6) + lic(C, T_{10}) = 5$ .

**Definition 2.** The *local itemset amount* is the sum of the local item amounts (i.e., the product of the local item count for a particular item purchased in a particular transaction and the item retail price) for all transactions which contain a particular item in a particular itemset, denoted as  $lisa_1(i, x)$ , where  $lisa_1(i, x) = \sum lia(i, t_k)$ ,  $lia(i, t)$  is the value at the intersection of row  $t$  and column  $i$  multiplied by the item retail price of item  $i$ ,  $i \in I$ ,  $x \subseteq I$ ,  $x \in t_k$ , and  $t_k \in D$ . Alternatively, the local itemset amount is the product of the local itemset count for a particular item in a particular itemset and the item retail price, denoted as  $lisa_2(i, x)$ , where  $lisa_2(i, x) = lisc(i, x) * irp(i)$ ,  $irp(i)$  is the item retail price,  $i \in I$ , and  $x \subseteq I$ .

**Query.** “Give the value of item  $C$  in itemset  $\{B, C\}$ .”

**Result.** The local itemset amount for item  $C$  in itemset  $\{B, C\}$  is  $lisa_1(C, \{B, C\}) = lia(C, T_3) + lia(C, T_6) + lia(C, T_{10}) = 25.00$ .

**Definition 3.** The *global itemset count* is the sum of the local itemset counts for all items in a particular itemset, denoted as  $gisc(x)$ , where  $gisc(x) = \sum lisc(i_k, x)$ ,  $x \subseteq I$ , and  $i_k \in x$ , for all  $k$ .

**Query.** “Give the quantity of all items in itemset  $\{B, C\}$ .”

**Result.** The global itemset count for itemset  $\{B, C\}$  is  $gisc(\{B, C\}) = lisc(B, \{B, C\}) + lisc(C, \{B, C\}) = 13$ .

**Definition 4.** The *global itemset amount* is the sum of the local itemset amounts for all items in a particular itemset, denoted as  $gis(a)(x)$ , where  $gis(a)(x) = \sum lisa_1(i_k, x)$ ,  $x \subseteq I$ , and  $i_k \in x$ , for all  $k$ , or alternatively,  $gis(a)(x) = \sum lisa_2(i_k, x)$ ,  $x \subseteq I$ , and  $i_k \in x$ , for all  $k$ .

**Query.** “Give the value of all items in itemset  $\{B, C\}$ .”

**Result.** The global itemset amount for itemset  $\{B, C\}$  is  $gis(a)(\{B, C\}) = lisa_2(B, \{B, C\}) + lisa_2(C, \{B, C\}) = 43.00$ .

**Definition 5.** The *total itemset count* is the sum of the global item counts (i.e., the sum of the local item counts for a particular item purchased in all transactions) for all items in a particular itemset, denoted as  $tisc(x)$ , where  $tisc(x) = \sum gic(i_k)$ ,  $gic(i)$  is the sum of all values in column  $i$ ,  $x \subseteq I$ , and  $i_k \in x$ .

**Query.** “Give the quantity of all items in the transaction database that are in itemset  $\{B, C\}$ .”

**Result.** The total itemset count for itemset  $\{B, C\}$  is  $tisc(\{B, C\}) = gic(B) + gic(C) = 25$ .

**Definition 6.** The *total itemset amount* is the sum of the global item amounts

(i.e., the sum of the local item amounts for a particular item purchased in all transactions) for all items in a particular itemset, denoted as  $tisa(x)$ , where  $tisa(x) = \sum gia(i_k)$ ,  $gia(i)$  is the value of item  $i$  in all transactions,  $x \subseteq I$ , and  $i_k \in x$ .

**Query.** “Give the value of all items in the transaction database that are in itemset  $\{B, C\}$ .”

**Result.** The total itemset amount for itemset  $\{B, C\}$  is  $tisa(\{B, C\}) = gia(B) + gia(C) = 92.00$ .

## 2.2 Share

We now introduce and define the notion of share in terms of the definitions from the previous section.

**Definition 7.** The *total item count local share* for a particular item in a particular itemset is the ratio of the local itemset count to the total item count (i.e., the sum of the global item counts for all items purchased in all transactions), expressed as a percentage, denoted as  $ticls(i, x)$ , where  $ticls(i, x) = (lisc(i, x)/tic) * 100$ ,  $tic$  is the quantity of all items in the transaction database,  $i \in I$ , and  $x \subseteq I$ .

**Query.** “Give the share of the quantity of item  $F$  in itemset  $\{D, F\}$  in relation to the quantity of all items in the transaction database.”

**Result.** The total item count local share for item  $F$  in itemset  $\{D, F\}$  is  $ticls(F, \{D, F\}) = (lisc(F, \{D, F\})/tic) * 100 = 10.6\%$ .

**Definition 8.** The *total item amount local share* for a particular item in a particular itemset is the ratio of the local itemset amount to the total item amount (i.e., the sum of the global item amounts for all items purchased in all transactions), expressed as a percentage, denoted as  $tials(i, x)$ , where  $tials(i, x) = (lisa_v(i, x)/tia) * 100$ ,  $tia$  is the total value of all items in the transaction database,  $i \in I$ ,  $x \subseteq I$ , and  $v \in \{1, 2\}$ .

**Query.** “Give the share of the value of item  $F$  in itemset  $\{D, F\}$  in relation to the value of all items in the transaction database.”

**Result.** The total item amount local share for item  $F$  in itemset  $\{D, F\}$  is  $tials(F, \{D, F\}) = (lisa_1(F, \{D, F\})/tia) * 100 = 15.9\%$ .

**Definition 9.** The *total item count global share* for a particular itemset is the ratio of the global itemset count to the total item count, expressed as a percentage, denoted as  $ticgs(x)$ , where  $ticgs(x) = (gisc(x)/tic) * 100$ ,  $x \subseteq I$ .

**Query.** “Give the share of the quantity of all items in itemset  $\{D, F\}$  in relation to the quantity of all items in the transaction database.”

**Result.** The total item count global share for itemset  $\{D, F\}$  is  $ticgs(\{D, F\}) = (gisc(\{D, F\})/tic) * 100 = 24.2\%$ .

**Definition 10.** The *total item amount global share* for a particular itemset is the ratio of the global itemset amount to the total item amount, expressed as a percentage, denoted as  $tiags(x)$ , where  $tiags(x) = (gis(x)/tia) * 100$ ,  $x \subseteq I$ .

**Query.** “Give the share of the value of all items in itemset  $\{D, F\}$  in relation to the value of all items in the transaction transaction database.”

**Result.** The total item amount global share for itemset  $\{D, F\}$  is  $tiags(\{D, F\}) = (gis(\{D, F\})/tia) * 100 = 28.9\%$ .

**Definition 11.** The *global itemset count local share* for a particular item in a particular itemset is the ratio of the local itemset count to the global itemset count, expressed as a percentage, denoted as  $giscls(i, x)$ , where  $giscls(i, x) = (lisc(i, x)/gisc(x)) * 100$ ,  $i \in I$ , and  $x \subseteq I$ .

**Query.** “Give the share of the quantity of item  $A$  in itemset  $\{A, D\}$  in relation to the quantity of all items in the itemset.”

**Result.** The global itemset count local share for item  $A$  in itemset  $\{A, D\}$  is  $giscls(A, \{A, D\}) = (lisc(A, \{A, D\})/gisc(\{A, D\})) * 100 = 46.2\%$ .

**Definition 12.** The *global itemset amount local share* for a particular item in a particular itemset is the ratio of the local itemset amount to the global itemset amount, expressed as a percentage, denoted as  $gisals(i, x)$ , where  $gisals(i, x) = (lisa_v(i, x)/gis(a(x)) * 100$ ,  $i \in I$ ,  $x \subseteq I$ , and  $v \in \{1, 2\}$ .

**Query.** “Give the share of the value of item  $A$  in itemset  $\{A, D\}$  in relation to the value of all items in the itemset.”

**Result.** The global itemset amount local share for item  $A$  in itemset  $\{A, D\}$  is  $gisals(A, \{A, D\}) = (lisa_1(A, \{A, D\})/gis(a(\{A, D\}))) * 100 = 21.3\%$ .

### 2.3 Frequent Itemsets

A frequent itemset was previously defined as an itemset whose support is greater than some user-specified minimum [2]. We now define frequent itemsets as used in the share-confidence framework.

**Definition 13.** An itemset is *locally frequent* if there is an item in the itemset such that at least one of the following conditions holds:

1. The total item count local share is greater than some user-specified minimum. That is,  $ticls(i_k, x) \geq minshare_1$ , where  $x \subseteq I$ ,  $i_k \in x$ , for some  $k$ , and  $minshare_1$  is the user-specified minimum share.
2. The total item amount local share is greater than some user-specified minimum. That is,  $tials(i_k, x) \geq minshare_2$ , where  $x \subseteq I$ ,  $i_k \in x$ , for some  $k$ , and  $minshare_2$  is the user-specified minimum share.

**Query.** “Give the frequent 2-itemsets whose local share for at least one item is at least 8%.”

**Result.** The locally frequent 2-itemsets are shown in Table 3. In Table 3, the *Itemset* column describes the items in the itemset, the *TIDs* column describes the transaction identifiers that contain the corresponding itemset, the  $ticls(i_1, x)$  and  $ticls(i_2, x)$  columns describe the total item count local share for items one and two, respectively, and the  $tials(i_1, x)$  and  $tials(i_2, x)$  columns describe the total item amount local share for items one and two, respectively.

**Table 3.** Locally frequent 2-itemsets

| <i>Itemset</i> | <i>TIDs</i>                                                        | <i>ticls(i<sub>1</sub>, x)</i><br>(%) | <i>ticls(i<sub>2</sub>, x)</i><br>(%) | <i>trials(i<sub>1</sub>, x)</i><br>(%) | <i>trials(i<sub>2</sub>, x)</i><br>(%) |
|----------------|--------------------------------------------------------------------|---------------------------------------|---------------------------------------|----------------------------------------|----------------------------------------|
| {A, D}         | T <sub>1</sub> , T <sub>7</sub> , T <sub>8</sub>                   | <b>9.09</b>                           | <b>10.6</b>                           | 2.73                                   | <b>10.1</b>                            |
| {B, E}         | T <sub>2</sub> , T <sub>6</sub> , T <sub>9</sub> , T <sub>10</sub> | <b>16.67</b>                          | 7.58                                  | 7.52                                   | <b>15.19</b>                           |
| {B, C}         | T <sub>3</sub> , T <sub>6</sub> , T <sub>10</sub>                  | <b>12.12</b>                          | 7.58                                  | 5.47                                   | 7.59                                   |
| {C, E}         | T <sub>4</sub> , T <sub>6</sub> , T <sub>10</sub>                  | <b>9.09</b>                           | 4.55                                  | <b>9.11</b>                            | <b>9.11</b>                            |
| {C, F}         | T <sub>4</sub> , T <sub>11</sub>                                   | <b>9.09</b>                           | 6.06                                  | <b>9.11</b>                            | <b>9.11</b>                            |
| {E, F}         | T <sub>4</sub> , T <sub>7</sub> , T <sub>9</sub>                   | 7.58                                  | <b>10.6</b>                           | <b>15.19</b>                           | <b>15.95</b>                           |
| {D, F}         | T <sub>5</sub> , T <sub>7</sub> , T <sub>8</sub>                   | <b>13.64</b>                          | <b>10.6</b>                           | <b>12.98</b>                           | <b>15.95</b>                           |
| {A, F}         | T <sub>7</sub> , T <sub>8</sub>                                    | 7.58                                  | <b>9.09</b>                           | 2.27                                   | <b>13.67</b>                           |
| {D, E}         | T <sub>7</sub>                                                     | 1.52                                  | 6.06                                  | 1.44                                   | <b>12.15</b>                           |

**Definition 14.** An itemset is *globally frequent* if every item in the itemset is locally frequent.

**Query.** “Give the frequent 2-itemsets whose local share for all items is at least 8%.”

**Result.** The globally frequent 2-itemsets are shown in Table 4. The columns in Table 4 have the same meaning as in Table 3.

**Table 4.** Globally frequent 2-itemsets

| <i>Itemset</i> | <i>TIDs</i>                                       | <i>ticls(i<sub>1</sub>, x)</i><br>(%) | <i>ticls(i<sub>2</sub>, x)</i><br>(%) | <i>trials(i<sub>1</sub>, x)</i><br>(%) | <i>trials(i<sub>2</sub>, x)</i><br>(%) |
|----------------|---------------------------------------------------|---------------------------------------|---------------------------------------|----------------------------------------|----------------------------------------|
| {A, D}         | T <sub>1</sub> , T <sub>7</sub> , T <sub>8</sub>  | <b>9.09</b>                           | <b>10.6</b>                           | 2.73                                   | <b>10.1</b>                            |
| {C, E}         | T <sub>4</sub> , T <sub>6</sub> , T <sub>10</sub> | 9.09                                  | 4.55                                  | <b>9.11</b>                            | <b>9.11</b>                            |
| {C, F}         | T <sub>4</sub> , T <sub>11</sub>                  | 9.09                                  | 6.06                                  | <b>9.11</b>                            | <b>9.11</b>                            |
| {E, F}         | T <sub>4</sub> , T <sub>7</sub> , T <sub>9</sub>  | 7.58                                  | 10.6                                  | <b>15.19</b>                           | <b>15.95</b>                           |
| {D, F}         | T <sub>5</sub> , T <sub>7</sub> , T <sub>8</sub>  | <b>13.64</b>                          | <b>10.6</b>                           | <b>12.98</b>                           | <b>15.95</b>                           |

## 2.4 Confidence

Confidence in an association rule  $X \Rightarrow Y$  was previously defined as the ratio of the number of transactions containing itemset  $X \cup Y$  to the number of transactions containing itemset  $X$  [2]. We now define confidence as used in the share-confidence framework.

**Definition 15.** The *count confidence* in an association rule  $X \Rightarrow Y$  is the ratio of the sum of the local itemset counts for all items in itemset  $X$  contained in  $X \cup Y$  to the global itemset count for itemset  $X$ , expressed as a percentage, denoted as  $cc(x, x \cup y)$ , where  $cc(x, x \cup y) = (\sum lisc(i_k, x \cup y) / gisc(x)) * 100$ ,  $x \subseteq I$ ,  $x \cup y \subseteq I$ , and  $i_k \in x$ , for all  $k$ .

**Query.** “Give the count confidence for the association rule  $\{B, C\} \Rightarrow \{E\}$ .”

**Result.** The count confidence for the association rule  $\{B, C\} \Rightarrow \{E\}$  is  $cc(\{B, C\}, \{B, C, E\}) = ((lisc(B, \{B, C, E\}) + lisc(C, \{B, C, E\})) / gisc(\{B, C\})) * 100 = 76.9\%$ .

**Definition 16.** The *amount confidence* in an association rule  $X \Rightarrow Y$  is the ratio of the sum of the local itemset amounts for all items in itemset  $X$  contained in

$X \cup Y$  to the global itemset amount for itemset  $X$ , expressed as a percentage, denoted as  $ac(x, x \cup y)$ , where  $ac(x, x \cup y) = (\sum lisa_v(i_k, x \cup y)/gis(a(x)) * 100$ ,  $x \subseteq I$ ,  $x \cup y \subseteq I$ ,  $i_k \in x$ , for all  $k$ , and  $v \in \{1, 2\}$ .

**Query.** “Give the amount confidence for the association rule  $\{B, C\} \Rightarrow \{E\}$ .”

**Result.** The amount confidence for the association rule  $\{B, C\} \Rightarrow \{E\}$  is  $ac(\{B, C\}, \{B, C, E\}) = ((lisa_2(B, \{B, C, E\}) + lisa_2(C, \{B, C, E\}))/gis(a(\{B, C\})) * 100 = 59.9\%$ .

### 3 Characterized Itemsets

#### 3.1 Example

We now present an example to demonstrate the *CI* algorithm and describe the primary data structures. In this example, let  $L_k^*$  and  $C_k^*$  denote the set of frequent itemsets from pass  $k$  and the set of candidate itemsets from pass  $k$ , respectively, and let  $R^*$  denote the relation containing the characterized itemsets. Each element of  $L_k^*$  and  $C_k^*$  contains three attributes: the itemset, the total item count local share, and the total item amount local share. Each element of  $R^*$  contains one attribute for each characteristic of interest and an attribute containing a list of all frequent itemsets sharing the corresponding characteristic attributes. Assume we are given the transaction database shown in Table 5. Also assume the user-specified minimum share is 15%. In Table 5, the column descriptions have the same meaning as the like-named columns in Table 1. Our task is to trace through the first three passes of *CI* to generate and store the characterized itemsets in  $R^*$ . For this example, we consider only the total item count local share to determine whether an itemset is frequent.

**Table 5.** A smaller example transaction database with counts

| TID   | A  | B | C  | D  | E |
|-------|----|---|----|----|---|
| $T_1$ | 1  | 2 | 5  | 0  | 0 |
| $T_2$ | 4  | 1 | 1  | 3  | 2 |
| $T_3$ | 3  | 0 | 2  | 1  | 0 |
| $T_4$ | 5  | 0 | 4  | 2  | 1 |
| $T_5$ | 2  | 3 | 3  | 4  | 0 |
|       | 15 | 6 | 15 | 10 | 3 |

After the first pass, *CI* generates  $L_1^*$  and  $R^*$  as shown in Tables 6 and 7, respectively. In Table 6, the *Itemset* column describes the items in each itemset and the *Share* column describes the total item count local share. In Table 7, the *Char. 1* and *Char. 2* columns describe the characteristics retrieved from the external database(s), and the *TIDs* column describes the transactions that share the corresponding characteristics (the TIDs are not actually stored in  $R^*$  and are merely shown here for reader convenience). The domain of the first and second characteristic is  $\{R, S\}$  and  $\{X, Y, Z\}$ , respectively.

After the second pass, *CI* generates  $L_2^*$  and updates  $R^*$  as shown in Tables 8 and 9, respectively. In Tables 8 and 9, the column descriptions have the same meaning as the like-named columns in Table 6 and Table 7, respectively. Also in

**Table 6.** Frequent itemsets contained in  $L_1^*$ 

| Itemset | Share (%) |
|---------|-----------|
| {A}     | 30.6      |
| {C}     | 30.6      |
| {D}     | 20.4      |

**Table 7.**  $R^*$  after the first pass

| Char. 1 | Char. 2 | TIDs       |
|---------|---------|------------|
| R       | X       | $T_1, T_4$ |
| S       | Y       | $T_2, T_5$ |
| S       | Z       | $T_3$      |

Table 9, the *Itemsets* column describes the frequent itemsets from the previous pass that share the identified characteristics.

**Table 8.** Frequent itemsets contained in  $L_2^*$ 

| Itemset | Share (%) |
|---------|-----------|
| {A, C}  | 61.2      |
| {A, D}  | 49.0      |
| {A, E}  | 24.5      |

After the third pass, *CI* generates  $L_3^*$  and updates  $R^*$  as shown in Tables 10 and 11, respectively. In Tables 10 and 11, the column descriptions have the same meaning as the like-named columns in Table 8 and Table 9, respectively.

The characterized itemsets in  $R^*$ , generated by *CI*, form a relation. In a relation, transforming a specific data description into a more general one is called generalization. Several algorithms have been proposed for finding generalized itemsets where concept hierarchies are used to classify items [10, 1]. Our approach differs from these in that we use concept hierarchies to classify the characteristic attributes. Fast and efficient implementations of *AOG* [7, 6, 13] are used to generate summaries where the characteristic attributes are generalized according to the concept hierarchies. If the concept hierarchies have relatively few levels (i.e., fewer than 10), and if multiple hierarchies are available for some attributes, the *AllGen* algorithm [12] is used to generate all possible summaries.

### 3.2 The *CI* Algorithm

In the description of *CI* that follows,  $L_k^*$ ,  $C_k^*$ , and  $R^*$  have the same meaning as in the example of the previous section. The  $k$ -th pass of the algorithm works as follows:

1. Repeat steps 2 to 5 until no new candidate itemsets are generated in pass  $(k - 1)$ .

**Table 9.**  $R^*$  after the second pass

| Char. 1 | Char. 2 | TIDs       | Itemsets                                       |
|---------|---------|------------|------------------------------------------------|
| R       | X       | $T_1, T_4$ | ( $\{A\}$ , 6), ( $\{C\}$ , 9), ( $\{D\}$ , 2) |
| S       | Y       | $T_2, T_5$ | ( $\{A\}$ , 6), ( $\{C\}$ , 4), ( $\{D\}$ , 7) |
| S       | Z       | $T_3$      | ( $\{A\}$ , 3), ( $\{C\}$ , 2), ( $\{D\}$ , 1) |

**Table 10.** Frequent itemsets contained in  $L_3^*$ 

| Itemset       | Share (%) |
|---------------|-----------|
| $\{A, C, D\}$ | 69.4      |
| $\{A, C, E\}$ | 34.7      |

2. Generate the candidate  $k$ -itemsets in  $C_k^*$  from the frequent  $(k - 1)$ -itemsets in  $L_{k-1}^*$  using the *Apriori* method described in [2, 5].
3. Partition the frequent  $(k - 1)$ -itemsets in  $L_{k-1}^*$  and update the candidate itemsets in  $C_k^*$ .
  - a. Repeat steps 3-b to 3-f until there are no more transactions to be retrieved from the database.
  - b. Retrieve the next transaction from the database.
  - c. Retrieve the corresponding characteristic tuple from  $R^*$ .
  - d. For each  $(k - 1)$ -itemset in the transaction, if it is contained in  $L_{k-1}^*$ , update the characteristic tuple.
    - (i) If itemset summary attributes already exist for this  $(k - 1)$ -itemset in the characteristic tuple, go to step 3-d-ii. step. Otherwise, create new itemset summary attributes in the characteristic tuple.
    - (ii) Increment the total quantity and total value attributes for this  $(k - 1)$ -itemset in the characteristic tuple.
  - e. If the characteristic tuple has been updated, save it in  $R^*$ .
  - f. For each  $k$ -itemset in the transaction, if it is contained in  $C_k^*$ , increment the associated total quantity and total value attributes.
4. Save the frequent  $k$ -itemsets in  $L_k^*$ .
  - a. Repeat steps 4-b and 4-c until there are no more itemset tuples in  $C_k^*$ .
  - b. Retrieve the next itemset tuple from  $C_k^*$ .
  - c. If the share of this itemset tuple is greater than the minimum specified, copy the itemset tuple to  $L_k^*$ .
5. Delete  $C_k^*$ .
6. Save  $R^*$ .

The first pass of the algorithm is a special pass which generates the frequent 1-itemsets and the characteristic relation, as follows:

1. Generate the candidate 1-itemsets in  $C_1^*$  and the characteristic relation  $R^*$ .
  - a. Repeat steps 1-b to 1-f until there are no more transactions to be retrieved from the database.
  - b. Retrieve the next transaction from the database.

**Table 11.**  $R^*$  after the third pass

| Char. 1 | Char. 2 | TIDs       | Itemsets                                                                                        |
|---------|---------|------------|-------------------------------------------------------------------------------------------------|
| R       | X       | $T_1, T_4$ | $\{\{A\}, 6\}, \{\{C\}, 9\}, \{\{D\}, 2\}, \{\{A, C\}, 15\}, \{\{A, D\}, 7\}, \{\{A, E\}, 6\}$  |
| S       | Y       | $T_2, T_5$ | $\{\{A\}, 6\}, \{\{C\}, 4\}, \{\{D\}, 7\}, \{\{A, C\}, 10\}, \{\{A, D\}, 13\}, \{\{A, E\}, 6\}$ |
| S       | Z       | $T_3$      | $\{\{A\}, 3\}, \{\{C\}, 2\}, \{\{D\}, 1\}, \{\{A, C\}, 5\}, \{\{A, D\}, 4\}$                    |

- c. For each 1-itemset in the transaction, if an itemset tuple already exists in  $C_1^*$ , go step 1-d. Otherwise, create a new itemset tuple in  $C_1^*$ .
  - d. For each 1-itemset in the transaction, increment the total quantity and total value attributes of the associated itemset tuple in  $C_1^*$ .
  - e. Using the appropriate key(s), retrieve the characterizing attributes for this transaction from the external database(s).
  - f. If a characteristic tuple containing these characteristics already exists in  $R^*$ , go step 1-b. Otherwise, create a new characteristic tuple in  $R^*$ .
2. Save the frequent 1-itemsets in  $L_1^*$ .
- a. Repeat steps 2-b and 2-c until there are no more itemset tuples in  $C_1^*$ .
  - b. Retrieve the next itemset tuple from  $C_1^*$ .
  - c. If the share of this itemset tuple is greater than the minimum specified, copy the itemset tuple to  $L_1^*$ .
3. Delete  $C_1^*$ .
4. Save  $R^*$ .

The running time and space requirements of  $CI$  are  $O(|c| * |t|)$  and  $O(|s|)$ , respectively, where  $|c|$  is the number of candidate itemsets in all iterations of the algorithm,  $|t|$  is the number of transactions, and  $|s|$  is the size of the largest candidate itemset in any pass.

## 4 Experimental Results

We ran all of our experiments on an IBM AT-compatible personal computer, consisting of a Pentium P166 processor with 64 MB of memory running Windows NT Workstation version 4.0. Input data was from a large database supplied by a commercial partner in the telecommunications industry. The database contained approximately 3.3 million tuples representing account activity for over 500 thousand customer accounts and 2200 unique items (identified by integers in the range  $[1 \dots 2200]$ ). Each tuple is either an equipment rental or service transaction containing the number of items and the cost of each item. An itemset was considered to be frequent if at least one of the following three conditions held: the minimum support was greater than 0.25%, the total item count global share was greater than 0.25%, or the total item amount global share was greater than 0.25%.

The 20 most frequent 1-itemsets ranked by support, total item count global share, and total item amount global share are shown in Figures 1, 2, and 3, respectively. In Figures 1 to 3, the first row of bars (i.e., those at the front of

the graph) corresponds to the total item amount global share (i.e., value), the second row corresponds to the total item count global share (i.e., quantity), and the third row corresponds to the support. The height of each bar corresponds to the percentage of share or support for the associated 1-itemset. There were 109 frequent 1-itemsets discovered.

Figure 1 shows that support over-represents the actual frequency with which a 1-itemset is purchased, in terms of both the quantity and value of the purchases. The support for the most frequent 1-itemset is approximately 25%, yet this itemset represents only approximately 5% of the total quantity of items purchased and only approximately 2% of the total value of items purchased. The ranking of these same 1-itemsets by total item count global share is similar to that of support, but the ranking by total item amount global share shows significant variation from both support and total item count global share.

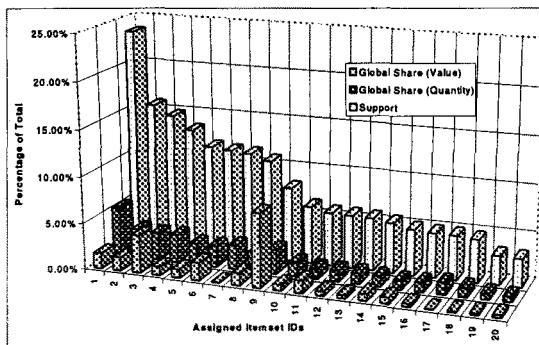


Fig. 1. 20 most frequent 1-itemsets ranked by support

Figure 2 shows that 14 of the frequent 1-itemsets that were ranked highest by support (i.e., those identified by integers less than or equal to 20), also appear in the 20 most frequent 1-itemsets ranked by total item count global share. The remaining six 1-itemsets (i.e., 101, 81, 25, 107, 100, 34) are shown to have a higher ranking when ranked by total item count global share. The 1-itemsets that include items 100, 101, and 107 are especially noteworthy since there were only 109 frequent 1-itemsets ranked. The support measure considers these items to be among the least important, yet when ranked by total item count global share, they are ranked eleventh, first, and eighth, respectively.

Figure 3 shows that nine of the frequent 1-itemsets that were ranked highest by support, also appear in the 20 most frequent 1-itemsets ranked by total item amount global share. It also shows that nine of the most frequent 1-itemsets which were ranked in the bottom 50% by support, are shown to be among the 20 most frequent when ranked by total item amount global share.

Similar results to those shown in Figures 1 to 3 were obtained when ranking  $k$ -itemsets. We present the results for 2-itemsets, shown in Table 12. Table 12 shows three sets of rankings for 2-itemsets, where each set contains three columns. In Table 12, the *Support*, *Share (Quantity)*, and *Share (Value)* columns

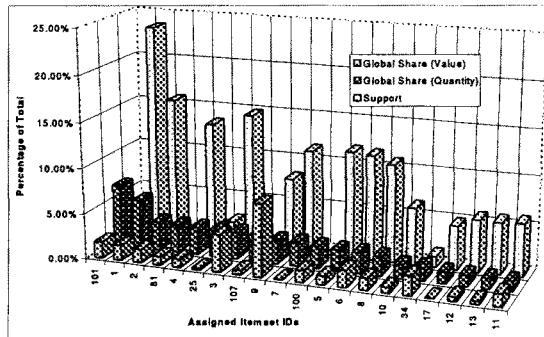


Fig. 2. 20 most frequent 1-itemsets ranked by total item count global share

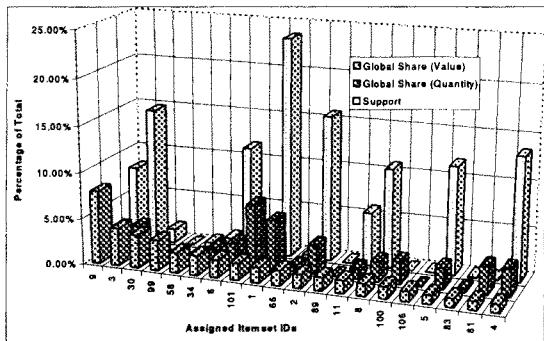


Fig. 3. 20 most frequent 1-itemsets ranked by total item amount global share

describe 10 itemsets ranked by support, total item count global share, and total item amount global share, respectively. In the first set, the first column shows the 10 most frequent 2-itemsets ranked by support. The second and third columns show the corresponding rank for these itemsets ranked by total item count and total item amount global share, respectively. In the second set, the second column shows the 10 most frequent 2-itemsets ranked by total item count global share. The first and third columns show the corresponding rank for these itemsets ranked by support and total item amount global share, respectively. In the third set, the third column shows the 10 most frequent 2-itemsets ranked by total item amount global share. The first and second columns show the corresponding rank for these itemsets ranked by support and total item count global share. There were 351 frequent 2-itemsets.

The 2-itemset ranked as most frequent by support (refer to the first set) and total item amount global share was ranked fourth by total item count global share. While this itemset does not represent the most frequent itemset sold in terms of the quantity of items, it was purchased in the greatest number of transactions and had the highest gross income of all 2-itemsets. In contrast, the

**Table 12.** 2-itemsets ranked by support and share

| Set 1 Rankings |                     |                  | Set 2 Rankings |                     |                  | Set 3 Rankings |                     |                  |
|----------------|---------------------|------------------|----------------|---------------------|------------------|----------------|---------------------|------------------|
| Support        | Share<br>(Quantity) | Share<br>(Value) | Support        | Share<br>(Quantity) | Share<br>(Value) | Support        | Share<br>(Quantity) | Share<br>(Value) |
| 1              | 4                   | 1                | 306            | 1                   | 18               | 1              | 4                   | 1                |
| 2              | 13                  | 3                | 341            | 2                   | 38               | 293            | 8                   | 2                |
| 3              | 17                  | 9                | 324            | 3                   | 27               | 2              | 13                  | 3                |
| 4              | 19                  | 12               | 1              | 4                   | 1                | 305            | 45                  | 4                |
| 5              | 20                  | 5                | 294            | 5                   | 23               | 5              | 20                  | 5                |
| 6              | 22                  | 11               | 316            | 6                   | 32               | 288            | 121                 | 6                |
| 7              | 27                  | 28               | 291            | 7                   | 24               | 75             | 80                  | 7                |
| 8              | 35                  | 33               | 293            | 8                   | 2                | 287            | 206                 | 8                |
| 9              | 47                  | 59               | 307            | 9                   | 29               | 3              | 17                  | 9                |
| 10             | 41                  | 109              | 301            | 10                  | 31               | 336            | 350                 | 10               |

2-itemset ranked tenth by support, for instance, was ranked 41-st by total item count global share and 109-th by total item amount global share. This itemset is ranked highly by support, yet its contribution to gross income is comparatively low.

The 2-itemset ranked as most frequent by total item count global share (refer to the second set) was ranked 306-th by support. This is an itemset where the items are typically purchased in multiples. Consequently, it is purchased more frequently than support seems to indicate. Similarly, 13 of the 15 most frequent 2-itemsets ranked highly by total item count global share are ranked below 291 by support.

The 2-itemset ranked tenth by total item amount global share (refer to the third set) was ranked 336-th by support and 350-th by total item count global share. The items in this itemset are relatively expensive items. Consequently, although not purchased as frequently as many other items, its contribution to gross income is comparatively high.

## 5 Conclusion

We have introduced the share-confidence framework for knowledge discovery from databases which classifies itemsets based upon characteristic attributes extracted from external databases. We suggested how characterized itemsets can be generalized according to concept hierarchies associated with the characteristic attributes. Experimental results demonstrated that the share-confidence framework can give more informative feedback than the support-confidence framework.

## References

1. R. Agrawal, K. Lin, H.S. Sawhney, and K. Shim. Fast similarity search in the presence of noise, scaling, and translation in time-series databases. In *Proceedings of the 21th International Conference on Very Large Databases (VLDB'95)*, Zurich, Switzerland, September 1995.

2. R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A.I. Verkamo. Fast discovery of association rules. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 307–328, Menlo Park, CA, 1996. AAAI Press/MIT Press.
3. R. Agrawal and J.C. Schafer. Parallel mining of association rules. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):962–969, December 1996.
4. S. Brin, R. Motwani, and C. Silverstein. Beyond market baskets: Generalizing association rules to correlations. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'97)*, pages 265–276, May 1997.
5. S. Brin, R. Motwani, J.D. Ullman, and S. Tsur. Dynamic itemset counting and implication rules for market basket data. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'97)*, pages 255–264, May 1997.
6. C.L. Carter and H.J. Hamilton. Efficient attribute-oriented algorithms for knowledge discovery from large databases. *IEEE Transactions on Knowledge and Data Engineering*. To appear.
7. C.L. Carter and H.J. Hamilton. Performance evaluation of attribute-oriented algorithms for knowledge discovery from databases. In *Proceedings of the Seventh IEEE International Conference on Tools with Artificial Intelligence (ICTAI'95)*, pages 486–489, Washington, D.C., November 1995.
8. C.L. Carter, H.J. Hamilton, and N. Cercone. Share-based measures for itemsets. In J. Komorowski and J. Zytkow, editors, *Proceedings of the First European Conference on the Principles of Data Mining and Knowledge Discovery (PKDD'97)*, pages 14–24, Trondheim, Norway, June 1997.
9. D.W. Cheung, A.W. Fu, and J. Han. Knowledge discovery in databases: a rule-based attribute-oriented approach. In *Lecture Notes in Artificial Intelligence, The 8th International Symposium on Methodologies for Intelligent Systems (ISMIS'94)*, pages 164–173, Charlotte, North Carolina, 1994.
10. J. Han and Y. Fu. Discovery of multiple-level association rules from large databases. In *Proceedings of the 1995 International Conference on Very Large Data Bases (VLDB'95)*, pages 420–431, September 1995.
11. J. Han and Y. Fu. Exploration of the power of attribute-oriented induction in data mining. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 399–421. AAAI/MIT Press, 1996.
12. R.J. Hilderman, H.J. Hamilton, R.J. Kowalchuk, and N. Cercone. Parallel knowledge discovery using domain generalization graphs. In J. Komorowski and J. Zytkow, editors, *Proceedings of the First European Conference on the Principles of Data Mining and Knowledge Discovery (PKDD'97)*, pages 25–35, Trondheim, Norway, June 1997.
13. H.-Y. Hwang and W.-C. Fu. Efficient algorithms for attribute-oriented induction. In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining (KDD'95)*, pages 168–173, Montreal, August 1995.
14. J.S. Park, M.-S. Chen, and P.S. Yu. An effective hash-based algorithm for mining association rules. *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'95)*, pages 175–186, May 1995.

# Automatic Visualization Method for Visual Data Mining

Yuichi Iizuka, Hisako Shiohara, Tetsuya Iizuka, and Seiji Isobe

NTT Information and Communication Systems Lab.

1-1 Hikari-no-oka, Yokosuka-shi

Kanagawa, 239 JAPAN

+81 468 59 2771

{iizuka, shiohara, tetsuya, isobe}@dq.isl.ntt.co.jp

**Abstract.** Data mining has drawn a great deal of attention as a technique for introducing effective business strategies. Data mining derives effective patterns and rules from a large amount of data to support the decision making process. Although various approaches have been attempted in some research fields, we focused on the visual data mining support system from the viewpoint of harnessing the perceptual and cognitive capabilities of the human user. In visual data mining support, defining the visualization model is important to obtain effective patterns. The main point of the definition is how to find useful data attributes as the visualization target. This paper proposes a visual data mining support system based on two key concepts: automatic target attribute selection method using a decision tree or correlation coefficient matrix and automatic scene definition generating methods. Application examples are presented and evaluated.

## 1 Introduction

The evolution of computing technology has enabled people to collect and store copious amounts of data from a wide range of sources. Through the various advancements in this evolution, modern database technology has been the facilitator in data storage. Despite these advancements, we still lack efficient data analysis technology. The market competition among enterprises intensifies everyday, emphasizing the importance of effectively using copious data so that decisive actions can be taken in business strategy. In such cases, data mining has drawn attention as a technique for the practical use of databases.

Many studies in the field of data mining have been completed using such techniques as statistics, machine learning, artificial intelligence, and database querying [1, 2, 3]. Almost all of these techniques are based on automatic rule induction algorithms [4, 5] and text-based result expressions. Because most of the results are presented in text form, it is difficult for analysts to efficiently and easily understand mining results.

On the other hand, visualization techniques such as scatter chart and histogram are used for general data analysis [6, 7]. For the data mining field, visualization was only used as a result-expression function such as to generate charts

for association rules or a decision tree. Using visualization in this way does not fully utilize its potential advantages in the data mining process. Our target is to achieve a visual data mining system. Features of the system are human-oriented data mining by applying perceptual and cognitive capabilities and capitalizing on the business knowledge and data analysis skill of the analysts. The key ideas of the system are an automatic target attribute selection method and a scene definition generation method. By introducing this system, users can immediately analyze visual results without going through a definition process and having to understand specific mining techniques.

This paper proposes a visual data mining system with two key concepts: automatic target attribute selection methods and scene definition generation methods. Some application examples are described and evaluated. Finally, the effectiveness of the system is discussed.

## 2 Visual Data Mining System

A visual data mining system is constructed by adding two functions to INFOVISER, a multi-dimensional data analysis tool employing a visualization model. We call the proposed visual data mining system, INFOVISER-MINE.

### 2.1 INFOVISER

The approach of multi-dimensional data analysis on INFOVISER is based on an original information visualization model called the Node-Line View Model [8, 9]. The visualization model can define node-type and line-type graphical objects as determined from a combination of layout and shape parameters based on analysis data. By introducing this model, INFOVISER provides a multi-dimensional representation of a data scene with options highlighting relationships among data such as association, cluster, tendency, comparison, and deviation.

INFOVISER has three main functions. The first is a data retrieval function that creates the subject data by retrieving information from source files and databases and then combines the data (Fig. 1 I1)). The second is a data conversion function that creates graphical information by converting character-based source data (Fig. 1 I2)). The third is a scenario definition function that selects the functions to highlight using a simple GUI tool (Fig. 1 I3)). In the data conversion scenario definition, users can select any attribute for mapping to one parameter of a plot profile such as axes, size, color, shape, and label. To adapt INFOVISER into a visual data mining system, automatic scenario definition and scene creation are essential.

### 2.2 Existing Visual Data Analysis Process

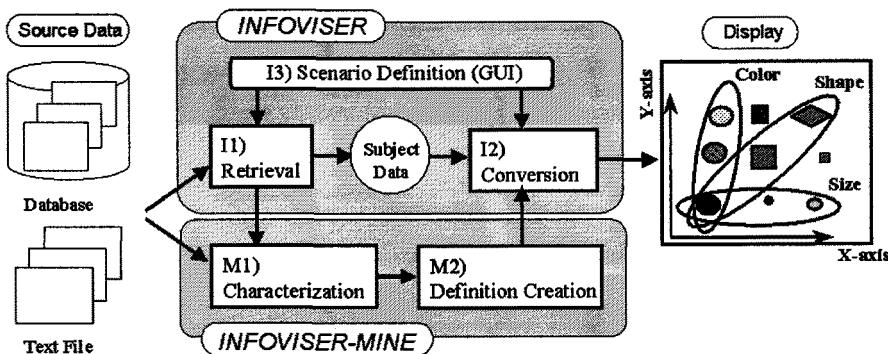
In general, data analysis by visualization comprises several steps: users analyze data by establishing a hypothesis, define a scene to visualize the data, and evaluate and verify the results. In visual data analysis, a data analysis cycle, from

establishing a hypothesis to verifying the results, is repeated on a trial and error basis. Establishing a hypothesis in visual data analysis means that users determine target attributes and allocate these attributes to plot profiles such as axes, size, color, shape, and label. Consequently, user can verify the degree correlation among target attributes through visualized results.

In visual data mining support, the system selects an attribute based on the extracted data characteristics, generates an information conversion definition, and presents visualization results to the user.

### 2.3 INFOVISER-MINE

INFOVISER-MINE adds two functions to INFOVISER to automatically create scenes. A characterization function automatically locates target attribute groups with a high correlation coefficient (Fig. 1 M1)). A definition creation function automatically creates a conversion definition using target attributes located by the characterization function (Fig. 1 M2)).



**Fig. 1.** Configure of INFOVISER and INFOVISER-MINE

### 3 Characterization of Data

The visual data mining support system presents to the user an effective visual pattern based on data characteristics. To accomplish this, it is important how the data characteristics are extracted and how these characteristics are used in selecting data attributes which are the target of the visualization model.

Though data mining techniques enable extracting various data characteristics, we focused on the kind of techniques which can extract relationships among

attributes. From such viewpoint, we attempt following two data mining techniques. One of the techniques is correlation coefficient matrix which is useful to detect relationships between two attributes. The other is decision tree [10] which is effective in searching attributes related for classifying records based on the value of target attribute. Attributes selection methods for visualize effective scenes using results of above two techniques are discussed below. Attributes selection Methods 1-3 are carried out using the correlation coefficient matrix and Method 4 is carried out using the decision tree.

### **3.1 Attributes Selection Method 1**

This attribute selection method focuses on the height of the correlation between two attributes based on a correlation coefficient matrix. Our aim is to show one example of the data characteristic to the user, as the initial stage to find patterns that exist inside the data. This method selects 2 attributes that have the maximum correlation coefficient and attributes that have a high degree of correlation to those attributes. The process after the correlation coefficient matrix is derived is:

1. Select 2 attributes ( $A_k, A_l$ ) that have maximum correlation coefficient ( $C_{kl}$ ).
2. Multiply correlation coefficient between attribute  $A_k$  and other attributes by those between attribute  $A_l$  and other attributes. ( $M_i = C_{ki} \times C_{li}$ )
3. Extract  $M$  in order from largest to smallest value except for  $M_k$  and  $M_l$ , and select attributes that are used to calculate  $M$ .

### **3.2 Attributes Selection Method 2**

This attribute selection method focuses on the correlation among the parts of all the attributes that belong to the data. Our aim is to select a user-specified number of attributes with a high degree of mutual correlation based on the limited number of dimensions that can be recognized by the user. The process after the correlation coefficient matrix is derived is:

1. The user-defined number of attributes to be selected is assumed to be  $m$ . Sum the correlation coefficients from the largest to  $m - 1$  for each attribute, and the calculated values are assumed to be  $S_i$ .
2. Select the attribute ( $A_k$ ) which has  $S_k$  (largest  $S_i$ ).
3. Select attributes that have a high degree of correlation to  $A_k$ .

### **3.3 Attributes Selection Method 3**

This attribute selection method focuses on the correlation among whole data attributes. The number of selected attributes is not limited. Our aim is to find the attributes that have a high degree of correlation among whole attributes, and show the tendency based on these attributes to the user. The process after the correlation coefficient matrix is derived is:

1. Sum correlation coefficients for each attributes. Calculated values are assumed to be  $S_i (S_i = C_{i1} + C_{i2} + \dots + C_{in})$ .
2. Select the attribute ( $A_k$ ) which has  $S_k$  (largest  $S_i$ ).
3. Select the attributes that have a high degree of correlation to  $A_k$ .

### 3.4 Attributes Selection Method 4

This attribute selection method employs a decision tree. Our aim is to find the attributes that influence the attribute that the user specified for the purpose of differentiation based on the characteristics of the decision tree. The process after generating the decision tree is derived is:

1. Extract node attributes in order of rank from high to low in the tree.
2. Exclude overlapping attributes in subordinate positions, and select attributes one by one.
3. Select the differentiation attribute that the user specified and used to generate the tree.

## 4 Definition Creation

The definition creation function automatically creates a conversion definitions using attributes selected by above methods. In this definitions, these attributes are mapped to parameters of a plot profile such as axes, size, color, shape, and label.

On the other hand, each parameter has own feature about representation capability and user perceptibility as follows.

1. In the case of scatter chart, users seem to be received strong impression in order of axes, color, size.
2. For representation of continuous value, gradation of color is more suitable than that of size.
3. Plot shape is negative to indicate wide value range.
4. Plot label is a helpful way to represent non-coded character value.

For this reason, attributes can not be mapped to all the parameters only by a priority obtained through the attributes selection method.

Our current definition creation function uses parameters such order as X-axis, Y-axis, color and size for automatic mapping. If the value of attribute mapped to any axis is discrete, priority of the attribute is exchanged with third one. In the case of attribute selection method 4, plot shape is also used to represent purpose of decision tree.

Also, user can change the mapping manually after examining automatically created scene through simple GUI.

## 5 Application Examples

This section evaluates the application of the attribute selection methods employing the correlation coefficient matrix and decision tree using an example taken from the medical field. Physical examinations and their visualization results form the basis of the evaluation.

**Table 1.** Correlation Coefficient Matrix

|                    | Age     | Sex    | Height | Weight       | Systolic pressure | Diastolic pressure | Cholesterol |
|--------------------|---------|--------|--------|--------------|-------------------|--------------------|-------------|
| Age                | 1.000   |        |        |              |                   |                    |             |
| Sex                | 0.094   | 1.000  |        |              |                   |                    |             |
| Height             | -0.105  | -0.742 | 1.000  |              |                   |                    |             |
| Weight             | 0.022   | -0.615 | 0.648  | 1.000        |                   |                    |             |
| Systolic pressure  | 0.055   | -0.043 | -0.067 | 0.296        | 1.000             |                    |             |
| Diastolic pressure | 0.067   | -0.224 | 0.041  | 0.358        | 0.879             | 1.000              |             |
| Cholesterol        | -0.081  | -0.180 | 0.096  | 0.446        | 0.443             | 0.440              | 1.000       |
| Glucose            | -0.057  | -0.243 | 0.130  | 0.272        | 0.202             | 0.261              | 0.396       |
| GOT                | -0.018  | -0.308 | 0.128  | 0.368        | 0.283             | 0.285              | 0.204       |
| GPT                | -0.019  | -0.326 | 0.147  | 0.470        | 0.258             | 0.308              | 0.185       |
| $\gamma$ GTP       | -0.078  | -0.307 | 0.161  | 0.333        | 0.196             | 0.238              | 0.308       |
| Obesity degree     | 0.086   | -0.230 | 0.063  | 0.743        | 0.504             | 0.512              | 0.600       |
| Drink              | -0.033  | -0.589 | 0.446  | 0.469        | 0.042             | 0.248              | 0.214       |
| Smoke              | -0.137  | -0.557 | 0.469  | 0.372        | 0.073             | 0.177              | 0.165       |
|                    | Glucose | GOT    | GPT    | $\gamma$ GTP | Obesity degree    | Drink              | Smoke       |
| Age                |         |        |        |              |                   |                    |             |
| Sex                |         |        |        |              |                   |                    |             |
| Height             |         |        |        |              |                   |                    |             |
| Systolic pressure  |         |        |        |              |                   |                    |             |
| Diastolic pressure |         |        |        |              |                   |                    |             |
| Cholesterol        |         |        |        |              |                   |                    |             |
| Glucose            | 1.000   |        |        |              |                   |                    |             |
| GOT                | 0.083   | 1.000  |        |              |                   |                    |             |
| GPT                | 0.218   | 0.842  | 1.000  |              |                   |                    |             |
| $\gamma$ GTP       | 0.271   | 0.709  | 0.657  | 1.000        |                   |                    |             |
| Obesity degree     | 0.268   | 0.397  | 0.485  | 0.334        | 1.000             |                    |             |
| Drink              | 0.201   | 0.169  | 0.214  | 0.242        | 0.265             | 1.000              |             |
| Smoke              | 0.132   | 0.212  | 0.203  | 0.238        | 0.197             | 0.465              | 1.000       |

The data used for the evaluation has 103 records and their attributes are "Age", "Sex", "Height", "Weight", "Systolic pressure", "Diastolic pressure", "Cholesterol", "Glucose", "GOT", "GPT", " $\gamma$ GTP", "Obesity degree", "Drink", and "Smoke". The correlation matrix derived from the data is shown in Table 1. The lists of attributes obtained from each attribute selection method are in Table 2. The obesity degree is set as the basis on which we differentiate the 103 records to generate a decision tree.

**Table 2.** Selected Attributes Using Methods 1-4

|               | Method 1           | Method 2       | Method 3       | Method 4     |
|---------------|--------------------|----------------|----------------|--------------|
| 1st attribute | Systolic pressure  | GPT            | Weight         | Cholesterol  |
| 2nd attribute | Diastolic pressure | GOT            | Obesity degree | Weight       |
| 3rd attribute | Obesity degree     | $\gamma$ GTP   | Height         | $\gamma$ GTP |
| 4th attribute | Cholesterol        | Obesity degree | Sex            | Age          |
| 5th attribute | Weight             | Weight         | GPT            | GOT          |

The results of visualization using attributes obtained by these methods are shown in the following. Basically, the first to forth attributes in Table 2 correspond to the X-axis, Y-axis, color, size, respectively. When applying method 4, the obesity degree attribute is mapped to the plot shape.

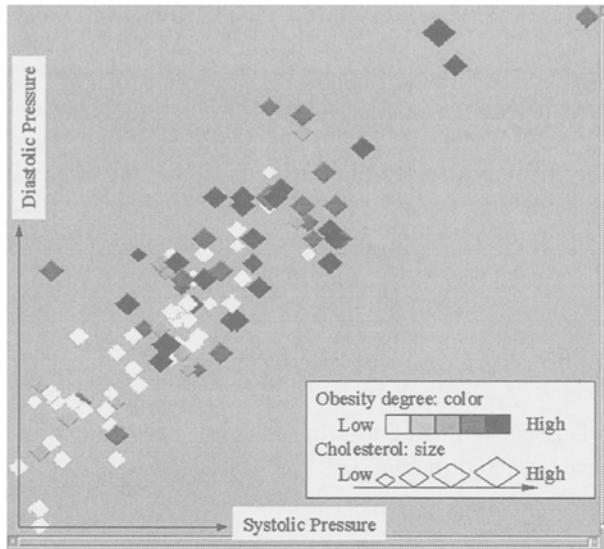
### 5.1 Result of Attribute Selection Method 1

The results of visualization using attributes written in the raw data of "Method 1" in Table 2, are depicted in Fig. 2.

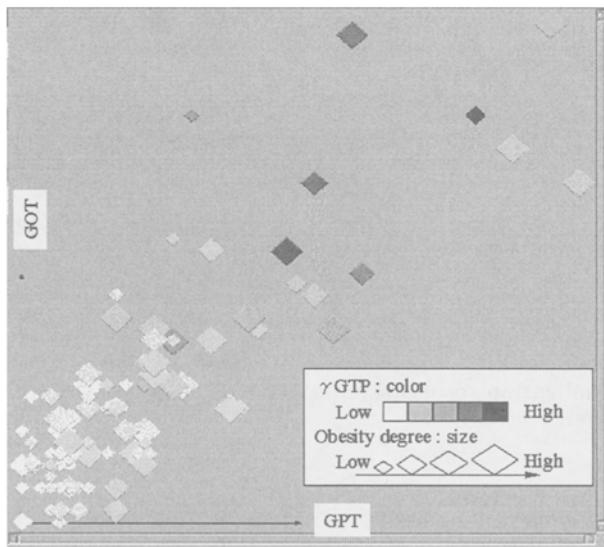
The distribution of values (height of correlation) can be understood by assigning attributes to the axes, color, and size. At a glance, the tendency can be understood. There is a high degree of correlation between systolic and diastolic pressure, (liner distribution), those who have hypertension, are generally overweight, (plots to the upper right are darker), but the correlation with the cholesterol level is not so high (plots of the same size are distributed uniformly). Moreover, values of attributes of each instance (each plot) can be shown. For example the following trend is assumed. The plots with low obesity-degree values and cholesterol in spite of high systolic and diastolic pressure measurements exists (light colored and small plots in the center). In this way, not only a general tendency to multiple attributes but also specific points, etc. can be recognized. Based on this, the analysis of the next stage can progress.

### 5.2 Result of Attribute Selection Method 2

The results of visualization using attributes written in the raw data of "Method 2" in Table 2, are depicted in Fig. 3.



**Fig. 2.** Visualization results using attributes obtained through method 1

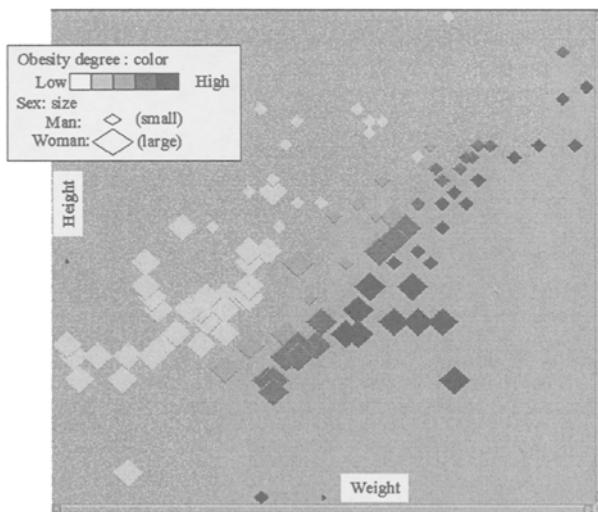


**Fig. 3.** Visualization results using attributes obtained through method 2

The relationship between layout and shape shows that GPT, GOT, and  $\gamma$ GTP have a high degree of correlation among them and these values have a tendency to be small for the plots with a low obesity degree. Especially, while the placement and color cover a wide range, the small plots are densely collected in the lower left part of the graph. The values of GOT are large compared to those of GPT for the dark shaded plots which indicate a large  $\gamma$ GTP value. Understanding the tendency can facilitate detailed analysis such as narrowing down the data for analysis or planning an analysis policy. For example, the analysis focusing on plots with large  $\gamma$ GTP values will be facilitated.

### 5.3 Result of Attribute Selection Method 3

The results of visualization using attributes written in the raw data of "Method 3" in Table 2, are depicted in Fig. 4.

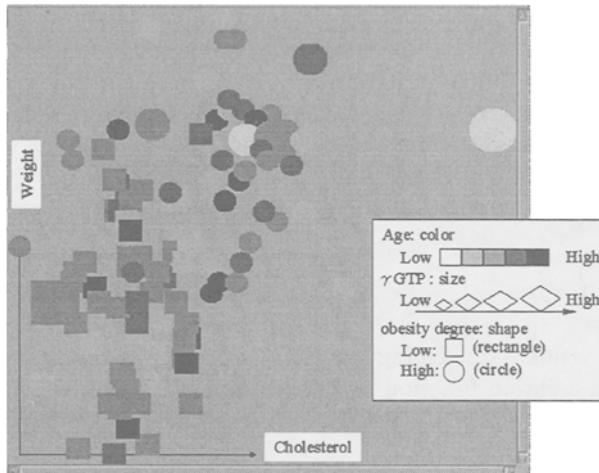


**Fig. 4.** Visualization results using attributes obtained through method 3

The obesity degree has second priority, but the value of this attribute is discrete. So, the obesity degree is mapped to the color of the plots, and the height (third priority) is mapped to the Y-axis. The results show that the attribute which has a high degree of correlation among all attributes is weight in the physical examination data. In addition, the definition of general obesity, such as a man with a high weight-to-height ratio has a high obesity-degree value, is shown intuitively.

#### 5.4 Result of Attribute Selection Method 4

The results of visualization using attributes written in the raw data of "Method 4" in Table 2, are depicted in Fig. 5.



**Fig. 5.** Visualization results using attributes obtained through method 4

The distribution state of the placement, size, and shape in the visualization results show that if weight and the cholesterol level are high, the obesity-degree value is high. In addition, it can be seen that the relationships between attributes other than the purpose of differentiation, for example cholesterol and weight have a low degree of correlation, though understanding such a relationship is not easy using only a decision tree.

### 6 Discussion

As the system decides effective visualization patterns based on the data characteristics using the above mentioned methods, we believe that these methods are effective in facilitating the establishment of a hypothesis when it is difficult to establish one.

Using a correlation coefficient matrix is effective when the user wants to know the relationship between two attributes. However, it is difficult to understand the correlation between multiple attributes, as the number of attributes increases. Methods 1-3 do not completely extract an effective correlation, but they do show an example of attributes that are correlated. Therefore, it seems that the generated visualization results are effective in understanding the data characteristics in the first stage of the analysis.

The decision tree is effective in understanding the specific conditions in detail. However, it is difficult to evaluate the specific patterns in the compound decision tree, because a detailed specification is done by looking sequentially at the classification result from a high-ranking vertex. The visualization representation using attributes obtained through Method 4 is effective in understanding the general tendency of the influence on the differentiation attribute. As described above, it is thought that displaying a multi-dimensional visualization model based on the data characteristics is useful in understanding the outline of data, and facilitates intuitive analysis. However, the effectiveness of the pattern presented differs depending on the analysis conditions such as the analysts, data, and background knowledge. Therefore, it is necessary to examine the method including interaction which considers the user's background knowledge. By doing so, the results may better satisfy the demand of individual users.

## 7 Conclusion

We proposed a visual data mining system by adding a characterization function and a definition creation function to INFOVISER. The system can aid analysts by facilitating the establishment of a hypothesis when it is difficult to establish one.

To realize the visual data mining system, we proposed two key ideas: an automatic target attribute selection method and a scene definition generation method. The automatic target attribute selection method is realized by adopting a correlation coefficient matrix and a decision tree. In an application to the medical field, more specifically to physical examination analysis, we confirmed the effectiveness of the system. In the future, selected target attributes must be filtered by considering the analyst's purpose.

## References

1. U. M. Fayyad and E. Simoudis, "Knowledge Discovery in Databases", Tutorial Notes, 14th International Joint Conference on Artificial Intelligence (IJCAI-95), 1995.
2. U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, "Advances in Knowledge Discovery and Data Mining", AAAI/MIT Press, 1995.
3. J. Han, "Data Mining Techniques", Tutorial notes, ACM-SIGMOD'96, 1996.
4. R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules", Proc. Of the 20th International conference on Very Large Data Bases, pp. 487-489, 1994.
5. T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama, "Mining Optimized Association Rules for Numeric Attributes", Proceedings of the 15th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, pp. 182-191, 1996.
6. D. A. Keim, "Database and Visualization", Tutorial Notes, ACM-SIGMOD'96, 1996.
7. S. G. Eick and D. E. Fyock, "Visualizing Corporate Data", AT&T Technical Journal, Vol. 75, No. 1, pp. 74-86, 1996.

8. K. Kurokawa, et al., "Information Visualization Environment for Character-based Database Systems", Proceedings of The First International Conference on Visual Information Systems, pp. 38-47, 1996.
9. K. Kurokawa and S. Isobe, "Data Visualization Model for Visual Knowledge Discovery in Databases", Workshop on Research Issues on Data Mining and Knowledge Discovery in cooperation with ACM-SIGMOD '97.
10. J. R. Quinlan, "C4.5: Programs for Machine Learning", Morgan Kaufmann Publishers, Inc., 1993.

# Rough-Set Inspired Approach to Knowledge Discovery in Business Databases

W. Kowalczyk<sup>1</sup> and Z. Piasta<sup>2</sup>

<sup>1</sup> Department of Mathematics and Computer Science  
Vrije Universiteit Amsterdam

De Boelelaan 1081A, 1081HV Amsterdam, The Netherlands  
e-mail: wojtek@cs.vu.nl

<sup>2</sup> Kielce University of Technology  
Mathematics Department

Al. 1000-lecia P.P. 5, 25-314 Kielce, Poland  
e-mail: zpiasta@sabat.tu.kielce.pl

**Abstract.** We present an approach to knowledge discovery in databases that is based on the idea of the attribute space partition. An inspiration for this approach was the methodology of the rough set theory. Two systems, *ProbRough* and *TRANCE*, which are representative of this approach are capable of inducing decision rules from databases with practically unlimited number of objects. The beam search strategy of *ProbRough* is guided by the global cost criterion and leads to inducing rough classifiers which are mainly intended for making decisions concerning new unseen objects. In case of *TRANCE*, the exhaustive search strategy in a space of user pre-specified models is guided by the criterion expressed in terms of local properties of the model. The relationships between values of attributes and decisions, detected by both systems, are presented in the form of rules that are easily understood by humans. The presented approach to knowledge discovery is illustrated on two real-world examples from database marketing. The rules induced by *ProbRough* and *TRANCE* provided a lot of useful information on customer behavior patterns and about the phenomenon of customer retention.

**Key words:** induction of rules, rough set theory, discretization of continuous data, noise handling, KDD applications in business

## 1 Introduction

Rapid market changes, increased competition, and new possibilities of information technology change the way companies make decisions about their businesses. Data warehouses give companies the ability to store information distributed across different computer systems and different business units in a consistent format. Operational data are integrated with customer, supplier, and market information. Corporate data can easily be accessed and managed. However, there is a gap between more powerful storage and retrieval systems and the business

professionals' ability to analyze the information contained in databases. This gap is successfully diminished by a new research area, data mining and knowledge discovery, which offers tools for extracting, in an automatic fashion, hidden patterns and relationships in databases (e.g., [2], [16], [10], [11]). The results of data mining and knowledge discovery can be then translated directly into business plans, considerably improving the quality of decisions.

The basic way of performing data mining and knowledge discovery is an application of supervised learning. In supervised learning a classification model is constructed inductively by exploring a number of known cases and generalizing from these cases. Thus, data mining may be identified with inducing decision rules, and a database plays the role of the training data. The rough set theory of Pawlak, [13], inspired many researchers to develop classification algorithms (see, for example, papers in [19], [21], [10], [20]). However, knowledge discovery in large databases is a novel area for the rough set methodology [1], which brings new challenges. For example, the key concept of rough sets, *reduct* (a minimal collection of relevant attributes), is no longer effectively computable (due to the size of data sets and the large number of attributes).

Recently, two approaches which avoid the use of reducts have emerged: *Rough Classifiers* developed by Lenarcik and Piasta, [7], [8], and *Rough Data Models*, introduced by Kowalczyk, [5], [4], [3]. Both approaches are focused on finding a relatively simple partition of the attribute space and then drawing some conclusions from the structure of this partition. Two systems which are representative for these approaches are *ProbRough*, see [14], [9], [15], and *TRANCE*, see [5]. In this paper we briefly describe both approaches and demonstrate their application to two real-world examples from database marketing.

## 2 Rough Classifiers and Rough Data Models

Let us assume that objects that have to be classified are described in terms of some attributes. Formally, let  $U$  denote the set of all objects ('cases' or 'patterns') and let  $A$  denote the set of attributes  $a_1, \dots, a_m$  defined on the universe  $U$ , i.e., every  $a_i$  is a function from  $U$  to a corresponding set of attribute values  $V_i$ ,  $a_i : U \rightarrow V_i$ , for  $i = 1, \dots, m$ . Moreover, let us assume that each object is assigned to one of  $n$  classes by a decision function  $d$  which takes values in the set  $D$  of decisions  $d_1, \dots, d_n$ , i.e.,  $d : U \rightarrow D$ . Frequently,  $D$  consists of two elements (e.g., 0 – *sale* and 1 – *no sale*), and cases are called 'positive' or 'negative' according to the values of  $d$ .

An attribute space is a collection of all  $m$ -tuples of values of attributes  $a_1, \dots, a_m$ . In other words, the attribute space is just a Cartesian product of sets of values of attributes  $a_1, \dots, a_m$ , i.e.,  $\mathcal{V} = V_1 \times \dots \times V_m$ .

In our further considerations we will concentrate on classification models which are determined by two components:

- a partition of the attribute space into a number of disjoint subsets, and
- a labeling of these subsets by elements of the set  $D$ .

The classification models are derived from a learning set: a finite set of learning objects with known values of attributes and decisions. Formally, the learning set is the triple

$$\mathbf{T} = (U', A, d),$$

where  $U' \subset U$ . This triple is often called an information system or a decision table, [13].

In our approach we will always assume that elements of the partition can be defined in terms of values of the attributes. Under this assumption the classification procedure is straightforward: to classify an element  $u \in U$  it is sufficient to find an element of the partition which contains  $u$  and use its label as a decision.

In practical situations we have to make some assumptions about the form of attributes and the form of elements of the partition. We assume that attribute values form a finite set of nominal values or are linearly ordered (e.g., real or integer numbers). The elements of partitions are then supposed to be Cartesian products of intervals (in case of ordered attributes) or arbitrary sets of values (in case of nominal attributes). To define rough classifiers more formally, we need some additional notions. By a segment in the value set  $V_i$  of the attribute  $a_i$  we mean the set  $\Delta_i \subset V_i$ , which is an interval, when the values of  $V_i$  are ordered, or an arbitrary subset, otherwise. By a feasible subset of  $\mathcal{V}$  we mean any set

$$\Delta = \Delta_1 \times \dots \times \Delta_m,$$

where  $\Delta_i$  is a segment in  $V_i$  ( $i = 1, \dots, m$ ).

A rough classifier is defined as a triple:

- (i) an attribute space  $\mathcal{V}$  with given value sets  $V_1, \dots, V_m$  of attributes, and a set of decisions  $D$ ,
- (ii) a finite partition  $\mathcal{S}$  of the space  $\mathcal{V}$  into disjoint feasible subsets,
- (iii) a mapping  $\kappa$  which assigns a decision  $\kappa(\Delta)$  to each element  $\Delta$  of the partition  $\mathcal{S}$ .

In the sequel, we identify the rough classifier with the mapping  $\kappa$ . The decision  $\kappa(\Delta)$  is the label of  $\Delta$ . For any  $\Delta = \Delta_1 \times \dots \times \Delta_m$  and its label  $\kappa(\Delta)$  we may write a decision rule:

$$\text{if } a_1 \in \Delta_1 \text{ and } \dots \text{ and } a_m \in \Delta_m \text{ then } \kappa(\Delta).$$

Sometimes we would like to assign several decisions to the same  $\Delta$  simultaneously, e.g., when  $\Delta$  contains the same number of ‘positive’ and ‘negative’ cases. In such situations  $\kappa(\Delta)$  should be defined as a subset of  $D$ , as described in [15]. In practice, a single decision is usually associated with each element of the partition.

Rough data models, RDMs, are rough classifiers with a linear ordering on all elements of the partition. The ordering is supposed to reflect problem specific importance of elements of the partition. The motivation behind introducing this ordering is the following: when a final model is developed we are often interested only in a (small) fragment of it—a part which is relevant to the given problem. For

example, if the goal of analyzing a marketing database is to select 5% of clients that are potentially most interested in a given product, then we are interested in performance of the model on these 5% of cases, neglecting the remaining 95%.

As we can see the difference between rough classifiers and rough data models is relatively small. However, the two systems that are based on these two concepts, *ProbRough* and *TRANCE*, operate in a different way and, as we will see in Section 3, produce different results.

Roughly speaking, *ProbRough* tries to find an optimal partition of the attribute space by minimizing the average misclassification cost, and then inducing some decision rules that describe the whole partition in a compact way. The cost criterion involves various costs of misclassifications and prior probabilities that reflect the distribution of the decisions in the universe. In many practical problems the capability to reduce the cost of misclassification rather than the number of misclassified objects is very important, [17]. In case of two decision classes, the costs and priors can be mutually compensated, [15]. As the search strategy, *ProbRough* uses a beam search which is guided by the cost criterion, [14].

*TRANCE* also tries to find partitions which optimize a certain criterion. This time, however, the objective function is usually expressed in terms of local properties of the resulting model (e.g., the highest classification rate on a fragment of the data set which contains at least 5% of all cases). Moreover, the search strategy is different: exhaustive search through a pre-specified collection of models. Usually, rough data models involve a small number of attributes, say, 2-5, with a small number of values, [3].

### 3 ProbRough and TRANCE

In this section we will describe the two systems which have been mentioned before: *ProbRough* and *TRANCE*. A more elaborate presentation of these systems can be found in [14] (*ProbRough*), and [3] (*TRANCE*).

#### 3.1 ProbRough

The algorithm ProbRough for rough classifier generation consists of two main phases: in the first phase we try to minimize the value of the cost criterion, while in the second phase – the number of decision rules.

**First phase of the algorithm.** In the first phase of the algorithm we obtain a global segmentation or a set of global segmentations of the attributes. By a segmentation of the attribute  $a_i$  we mean a partition of the value set  $V_i$  into disjoint segments  $\Delta_i^{(1)}, \dots, \Delta_i^{(s_i)}$ . By a global segmentation we mean a family of segmentations: each element of this family is a segmentation of one attribute. Any global segmentation uniquely determines a partition of the whole attribute space into the feasible subsets

$$\Delta_1^{(j_1)} \times \dots \times \Delta_m^{(j_m)}, \quad 1 \leq j_1 \leq s_1, \dots, 1 \leq j_m \leq s_m.$$

Obviously, the number of elements of this partition is equal to  $s_1 \dots s_m$ . By a value of the criterion for a global segmentation we mean the value of the criterion obtained for the corresponding partition.

We find a global segmentation or a set of global segmentations by using a refinement procedure which is iterated until a termination condition is met. A division process of a global segmentation is the main element of this procedure. By a division of a global segmentation we mean a division of one segment of this segmentation into a pair of new segments. The division leads to a new global segmentation and a new value of the criterion. In each iteration of the procedure all the input global segmentations are divided and a new set of output global segmentations is obtained. In the first iteration the input segmentation consists of the whole value sets  $V_i$  as the segments. The output of the first iteration is the input of the second one, and so on. In each iteration we accept only these divisions that lead to the decrease of the minimum criterion value obtained in the previous iteration. The number of the best criterion values associated with the accepted divisions is determined by the beam width of the search. The procedure can be terminated before reaching a prespecified maximum number of iterations when each new division does not lead to the decrease of the minimum value of the criterion so far reached. In such a case, the set of global segmentations generated in the last iteration reached is the final result of the procedure. The number of iterations in the first phase of the algorithm may be given in advance or can be optimized. In [14], we use the  $k$ -fold cross-validation method for this purpose.

**Second phase of the algorithm.** In the second phase of the algorithm only these global segmentations participate which are associated with the minimum criterion value obtained in the first phase. The criterion value remains unchanged during this phase of the algorithm. Below we present the process of rough classifier generation from one of the resulting partitions of the attribute space obtained in the first phase.

We start from generating all feasible subsets  $\Delta \subset \mathcal{V}$  which can be represented as unions of the partition elements with a common admissible (majority) decision. A label for any such  $\Delta$  is the intersection of labels associated with the components. A label is assigned to the subset  $\Delta$  in an automatic fashion. Therefore, the partition element  $\Delta$  can be identified with the rule. By volume of  $\Delta$  we mean the number of elements of the resulting partition which are included in  $\Delta$ .

Next, we order the generated rules. Bigger values of the criterion used correspond to rules with one well distinguished admissible decision. For two rules with the same value of the criterion, the better one is the rule with the bigger volume. If this criterion is also not decisive, then we count the rules as equivalent. We take the best subset  $\Delta$  as the first rule. In each next step we select the best rule that is disjoint with the previous one. We stop this process when the whole attribute space  $\mathcal{V}$  is filled up with the disjoint rules. If in any step of this process there is a possibility to select some equivalent rules (a case of ties), then

we consider all the equivalent variants. Finally, we choose the rule-sets with the minimum number of rules as the resultant set of rough classifiers.

### 3.2 TRANCE

The key feature of rough data models is their computational simplicity. A procedure which generates and evaluates a single model takes  $O(n)$  steps, where  $n$  is the number of records. In practice it means that for data sets which fit into computer main memory RDMs can be generated very fast (from a few milliseconds to a few seconds, depending on the size of the data set and computer speed). Therefore, it is possible to explore relatively big collections of RDMs to find the one which optimizes a predefined performance measure.

*TRANCE*, a Tool for Rough data ANalysis, Classification, and clustEring, is a system that supports the process of constructing and evaluating rough data models. Basically, it can be used for processing numeric-valued data tables. *TRANCE* contains a number of modules that cover different stages of the model construction process. The Data Pre-Processing Module contains numerous procedures that are useful for data pre-processing: procedures for removing outliers, for discretizing attributes, for statistical data analysis, for processing time series (rescaling, smoothing, aggregation), etc. The central part of the system is the Search Module. It is responsible for searching through the pre-defined space of models for an optimal one. Results of this module are passed to the Model Evaluation Module which contains procedures that calculate and plot various performance measures. Finally, the Rule Extraction Module generates rules from the best model.

The system is suitable for processing huge data sets with millions of records. It has been successfully applied in the field of marketing and finance to tasks such as: mailing selection, fraud detection, modelling customer behaviour, retention, and attrition.

The process of constructing a model for the given data set involves a number of choices: selection of attributes, their discretization (in case of continuous attributes), various transformations, etc. Usually, the exploration of numerous alternatives is prohibited by the time complexity of the involved algorithms. In case of RDMs the situation is different: we can generate them very quickly and therefore consider many scenarios. To be more specific, we will illustrate this approach on two problems: identification of important combinations of attributes and simultaneous discretization of several attributes.

**Selection of important attributes.** There are many methods for estimating the importance of a single attribute. For example, one can calculate various statistical measures of dependency, or, more often, information gain, [18]. Unfortunately, these methods fail if one wants to build a model which is based on a combination of several attributes. For example, it may happen that a combination of two unimportant attributes may result in a very good model, or two attributes that were found to be important may depend on each other and combining them would add almost nothing.

Using RDMs one may systematically search through models which are built on all combinations of 2-5 attributes. Such an approach is feasible if the original data set involves 10-100 attributes with very few values (2-3). If the number of attributes is higher, one may think about an application of a local search procedure, which, for example, starts with an empty set of attributes, incrementally adding attributes that lead to the highest increase of performance of the corresponding RDM. Continuous attributes should be discretized into 2-5 intervals. Because there are many discretization methods available, with no clear "winner", [12], a number of alternatives could be tried.

**Global discretization of continuous attributes.** Using RDMs we may directly search for discretizations which optimize the objective function. The procedure that we successfully used in [4] works as follows. First, we have to identify a few attributes which will be used in the final model. For this purpose we may use the strategy described above: discretize all continuous attributes into a small number of intervals with an *ad hoc* selected discretization method and then try all combinations of 2-5 attributes. When such a collection of attributes is found we discretize each of them into, say, 10 intervals and consider the resulting splitting points (9 for each attribute) as potential splitting points of the final discretization. Because this discretization will involve, say, 1-2 splitting points (per attribute), we can systematically try all combinations of 1 or 2 element subsets of 9- element sets (for each attribute) to find the best selection. Note that for a single attribute there are  $9 + 36 = 45$  possible combinations, so for a group of 3 attributes we will have to generate  $45 \cdot 45 \cdot 45 = 91125$  models – a number which is still acceptable. Clearly, the resulting discretization might be not the optimal one (due to the "rough" choice of the 9 starting points), but in practice this approach works quite well.

## 4 Two case studies

The presented approach to knowledge discovery is illustrated on two real-world examples from database marketing. Database marketing is a method of searching for patterns in customer preferences by using stored information, such as credit card purchase history, past promotion data, customer sales and member tracking data. Identified patterns and their associated customer profiles are helpful in implementing a promotion which is likely to influence purchase decisions and increase sales. Customer behavior patterns are also useful in formulating marketing, sales and customer support strategies.

### 4.1 Direct marketing

The first example concerns a large customer database. *ProbRough* and *TRANCE* were used in order to identify the characteristics of customers who are likely to respond to an offer or reject it. The database consists of 95,222 objects. The objects are characterized by 28 attributes reflecting social and economic status

of the customers as well as customers purchase history. The attributes are of mixed type: from unordered qualitative ones, e.g., marital status, occupation, credit card indicator, to continuous quantitative ones, e.g., customer age, home market value, length of residence. There are 19 attributes with missing values. Some of these attributes have even more than 50% of unknown or inapplicable values. Each object is assigned to one of the two decision classes: *sale* or *no sale*. The class *sale* is represented by 47,190 objects, while *no sale* by 48,032 objects. The prior probabilities were assumed to be equal to the observed frequencies of the decisions in the database. The misclassification cost matrix was defined in the standard form which is equivalent to minimizing the average number of misclassified objects. Because of a large number of missing values in the data, a minimum percentage of learning objects that has to be used in computing the criterion value was assumed as 50%, [15]. The best model generated by the *ProbRough* system consists of the following decision rules:

1. if  $x_{21} \in \{5, 6, 7, 8, 9\}$  and  $x_{22} \in \{D, E, \dots, S\}$  then  $d = \text{no sale}$ ,
2. if  $x_{19} \geq 33$  and  $x_{21} \in \{1, 2, 3, 4\}$  then  $d = \text{sale}$ ,
3. if  $x_{19} \geq 33$  and  $x_{21} \in \{5, 6, 7, 8, 9\}$  and  $x_{22} \in \{A, B, C\}$  then  $d = \text{sale}$ ,
4. if  $x_{19} < 33$  and  $x_{21} \in \{1, 2, 3, 4\}$  and  $x_{22} \in \{D, E, \dots, S\}$  then  $d = \text{no sale}$ ,
5. if  $x_{19} < 33$  and  $x_{21} \in \{5, 6, 7, 8, 9\}$  and  $x_{22} \in \{A, B, C\}$  then  $d = \text{no sale}$ ,
6. if  $x_{19} < 33$  and  $x_{21} \in \{1, 2, 3, 4\}$  and  $x_{22} \in \{A, B, C\}$  then  $d = \text{sale}$ .

The rules can be easily read. For example, the three strongest rules can be formulated as follows:

1. if estimated income is \$50,000 or more, and home market value is \$75,000 or more, then *no sale*;
2. if customer age is at least 33, and estimated income is under \$50,000, then *sale*;
3. if customer age is at least 33, and estimated income is \$50,000 or more, and home market value is under \$75,000, then *sale*.

The resulting optimal collection of the 6 decision rules can be used in order to assign any new customer to the *sale* or *no sale* class. Clearly, we are most interested in rules which have a high ratio of *sale* objects to *no sale* objects. Unfortunately, these ratios are not very impressive.

The *TRANCE* system performed systematic search with the same objective function (minimization of the error rate) over the space determined by the three most relevant attributes identified by *ProbRough*, and produced a similar result: the estimated misclassification error was almost the same and rather high. However, the situation changed when *TRANCE* was asked to generate ‘interesting rules’: rules that have a relatively high coverage and the percentage of objects that belong to the decision class *sale* is as high as possible. Clearly, this is a two-objective optimization problem: we want to maximize two (conflicting) performance measures at the same time. To cope with this problem we proceeded as follows: we tried to find rules that cover maximum percentage of interesting

objects, subject to a constraint that each rule should cover at least 3000, 4000, or 5000 cases (thus we run our system 3 times). Rules found in this way had relatively high percentage of objects from the *sale* class (64.22%, 63.23%, 62.45%, respectively). The best three rules look as follows ('?' denotes 'unknown'):

1. if  $x_{19} \in \{14, \dots, 31\}$  and  $x_{21} \in \{1\}$  and  $x_{22} \in \{?, A, B\}$  then  $d = \text{sale}$   
*sale to no sale ratio* = 2014 : 1122.
2. if  $x_{19} \in \{10, \dots, 32\}$  and  $x_{21} \in \{1, 2\}$  and  $x_{22} \in \{?, A, B\}$  then  $d = \text{sale}$   
*sale to no sale ratio* = 2558 : 1487.
3. if  $x_{19} \in \{13, \dots, 31\}$  and  $x_{21} \in \{1, 2, 3\}$  and  $x_{22} \in \{?, A, B\}$  then  $d = \text{sale}$   
*sale to no sale ratio* = 3350 : 2014.

As we can see, in spite of the fact that apparently the relationships between the values of the attributes used for customer characterization and the decisions are not strong we were able to generate some useful rules that cover a fragment of our universe.

## 4.2 Customer retention

One of the objectives of a mutual fund investment company is to increase its value. It can be achieved, for example, by increasing the cash flow, or, more specifically, by increasing the cash inflow (acquisition) and reducing the cash outflow (retention). For a healthy growth acquisition and retention efforts have to be brought into balance. The cash outflow can be reduced, for example, by preventing clients from quitting their relationship with the company. A climbing defection rate is namely a sure predictor of a diminishing flow of cash from customers to the company – even if the company replaces the lost customers – because older customers tend to produce greater cash flow and profits. They are less sensitive to price, they bring along new customers, and they do not require any acquisition or start-up costs. In some industries, reducing customer defections by as little as five percents can double profits. Customer retention is therefore an important issue. To be able to increase customer retention the company has to be able to predict which clients have a higher probability of defecting. It is also necessary to know what distinguishes a stopper from a non-stopper, especially with respect to characteristics which can be influenced by the company. Given this knowledge the company may focus their actions on the clients which are the most likely to defect, for example, by providing them extra advice and assistance.

In our research, [4], which was carried out in cooperation with a big mutual fund investment company, we tried to discover some factors (or their combinations) which discriminate between *stoppers* and *non-stoppers*, and which are early warnings for customer defection. As a starting point for our investigations we have used a fragment (about 15.000 cases, each case characterized by a few

hundred values) of a database containing information about more than 500.000 clients. Conceptual analysis of the available data led to a reduction of the number of available attributes to 212. To get some idea about the importance and relationships between various attributes a number of standard tests were carried out. First of all, we have generated numerous plots which are routinely used in statistical data analysis: frequency histograms, means, density estimates, etc. In order to identify most important attributes we have calculated, for every attribute, values of three ‘importance measures’: correlation coefficients, coefficients of concordance and information gain. All three measures provided similar results and allowed us to identify 9 attributes with values strongly related with the decisions. Given these 9 attributes we have used both systems, *TRANCE* and *ProbRough*, to build some models.

The *TRANCE* system has generated a few million models and automatically selected a number of most interesting ones. Further, by focusing on these best models a number of interesting rules were induced. As a side effect of this analysis it turned out that there were only 4 strongly influential variables and the remaining variables could be skipped. Models were analyzed in terms of their overall accuracy (classification error rate), behaviour (response curves), and complexity (the number and structure of corresponding partitions).

The *ProbRough* system was used to generate a number of rough classifiers. In contrast to models generated with *TRANCE* which were focused on local properties of the models, *ProbRough* induced models that were oriented on optimizing a global criterion (average misclassification cost). As a result, rules obtained by *ProbRough* were less focused on identifying potential stoppers and more on ‘getting a global picture’: inducing an optimal collection of rules in order to assign any new customer to the *stopper* or *non-stopper* class. However, some rules discovered by *ProbRough* had a very good ratio of *stoppers* to *non-stoppers*. The estimated classification error rate was much lower than in the first case study. This means that now the attributes were more relevant to the discovery task. Piatetsky-Shapiro, [16], observes that no amount of data will allow prediction based on attributes that do not capture the required information. It is worth stressing that the best linear discriminant classifier derived from the customer retention database classified unseen cases with significantly higher error rate than the rough classifier.

During both experiments, performed with the *TRANCE* and *ProbRough* systems, a number of significant rules which characterize various groups of clients have been found. These rules provided a lot of useful information about the phenomenon of retention.

## 5 Conclusions

The main contribution of our paper lies in presenting an efficient and simple approach to knowledge discovery in databases. The systems, *ProbRough* and *TRANCE*, are capable of generating decision rules from large datasets with practically unlimited numbers of objects. In case of *ProbRough* this capability is

obtained by a special implementation of the algorithm which does not require that the dataset is kept in main memory, [15]. *ProbRough* accepts databases with noisy and inconsistent information, delivered by an arbitrary number of attributes of any type. Continuous attributes are directly incorporated by the algorithm. The data may have a great number of missing values. Moreover, *ProbRough* enables the use of background knowledge in the form of prior probabilities of decisions and different costs of misclassification. The resulting rough classifiers are not sensitive to outliers in the data. The domains of induced decision rules are disjunctive and fill up the whole attribute space. As a result, any new object is covered by exactly one rule. Any strong and decisive rule induced by *ProbRough* can be used in isolation from the remaining part of the generated classifier. The *TRANCE* system is capable of deriving decision rules which are optimal in terms of local properties of the resulting model. In many practical problems we are interested only in a part of a final model which is relevant to the given problem. Knowledge that is discovered by using the *ProbRough* and *TRANCE* systems enables us to understand the given problem better and to explain the circumstances of the decisions. A representation of the learned knowledge is transparent and allows the user to discover new facts which explain the nature of the phenomena under consideration.

## 6 Acknowledgments

This work has been supported by grant no. 8 T11C 010 12 from the State Committee for Scientific Research (KBN). We thank Dr. W. Ziarko for providing the direct marketing customer database.

## References

1. Deogun, J. S., Raghavan, V. V., Sarkar, A. and Sever, H. (1997). Data mining: trends in research and developments. In: Lin, T. Y. and Cercone, N. (eds), *Rough Sets and Data Mining*, Kluwer Academic Publishers, Boston/ London/ Dordrecht, pp. 9–45.
2. Fayyad, U. M., Piatetsky-Shapiro, G., Smyth, P. and Uthurusamy, R. (1996). *Advances in Knowledge Discovery and Data Mining*, AAAI/MIT Press, London.
3. Kowalczyk, W. (1998). Rough Data Modelling: a new technique for analyzing data. In: L. Polkowski and A. Skowron (eds.) *Rough Sets in Knowledge Discovery*, Physica–Verlag (to appear).
4. Kowalczyk, W. and Slisser, F., (1997). Analyzing customer retention with rough data models. In *Proceedings of the 1st European Symposium on Principles of Data Mining and Knowledge Discovery*, PKDD'97, Trondheim, Norway, Lecture Notes in AI 1263, Springer, pp. 4–13.
5. Kowalczyk, W. (1996). TRANCE: A tool for rough data analysis, classification and clustering. In S. Tsumoto, S. Kobayashi, T. Yokomori, H. Tanaka and A. Nakamura (eds.), *Proceedings of the Fourth International Workshop on Rough Sets, Fuzzy Sets, and Machine Discovery*, RSFD'96, Tokyo University, pp. 269–275.

6. Kowalczyk, W. (1996). Analyzing temporal patterns with rough sets. In *Proceedings of the Fourth European Congress on Intelligent Techniques and Soft Computing, EUFIT'96*, Volume I, Aachen, Germany, pp. 139–143.
7. Lenarcik, A. and Piasta, Z. (1994). Rough classifiers. In: Ziarko, W. (ed.), *Rough Sets, Fuzzy Sets and Knowledge Discovery*, Springer-Verlag, London, pp. 298–316.
8. Lenarcik, A. and Piasta, Z. (1994). Deterministic rough classifiers. In *Conference Proceedings of the Third International Workshop on Rough Sets and Soft Computing*, San Jose State University, CA, pp. 434–441.
9. Lenarcik, A. and Piasta, Z. (1997). Probabilistic rough classifiers with mixtures of discrete and continuous condition attributes. In: Lin, T. Y. and Cercone, N. (eds), *Rough Sets and Data Mining*, Kluwer Academic Publishers, Boston/ London/ Dordrecht, pp. 373–383.
10. Lin, T. Y. and Wildberger, A. M., (eds) (1995). *Soft Computing: Rough Sets, Fuzzy Logic, Neural Networks, Uncertainty Management and Knowledge Discovery*, Simulation Councils, Inc., San Diego CA.
11. Michalski, R. S., Bratko, I. and Kubat, M. (eds) (1997), *Machine Learning and Data Mining: Methods and Applications*, Wiley, New York.
12. Nguyen, H.S. and Nguyen, S.H. (1998). Discretization Methods in Data Mining. In: L. Polkowski and A. Skowron (eds.) *Rough Sets in Knowledge Discovery*, Physica-Verlag (to appear).
13. Pawlak, Z. (1991). *Rough Sets: Theoretical Aspects of Reasoning About Data*, Kluwer Academic Publishers, Dordrecht.
14. Piasta, Z. and Lenarcik, A. (1996). Rule induction with probabilistic rough classifiers, ICS Research Report 24/96, Warsaw University of Technology, to appear in *Machine Learning*.
15. Piasta, Z. and Lenarcik, A. (1998). Learning rough classifiers from large databases with missing values. In: L. Polkowski and A. Skowron (eds.) *Rough Sets in Knowledge Discovery*, Physica-Verlag (to appear).
16. Piatetsky-Shapiro, G. (1996). Data mining and knowledge discovery in business databases, In: Ras, Z. W. and Michalewicz M.(eds), *Foundations of Intelligent Systems*, Springer-Verlag, Berlin Heidelberg, pp. 56–67.
17. Provost, F. and Fawcett, T. (1997). Analysis and visualization of classifier performance: comparison under imprecise class and cost distributions. In *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining*, AAAI Press.
18. Quinlan, R. (1986). Induction of decision trees, *Machine Learning* 1, 81–106.
19. Slowinski, R. (ed.) (1992). *Intelligent Decision Support: Handbook of Applications and Advances of the Rough Sets Theory*, Kluwer Academic Publishers, Dordrecht.
20. Tsumoto, S., Kobayashi, S., Yokomori, T., Tanaka, H and Nakamura, A., (1996) *Proceedings of the Fourth International Workshop on Rough Sets, Fuzzy Sets, and Machine Discovery*, University of Tokyo.
21. Ziarko, W., (ed.) (1994). *Rough Sets, Fuzzy Sets and Knowledge Discovery*, Springer-Verlag, London.

# Representative Association Rules

Marzena Kryszkiewicz

Institute of Computer Science  
Warsaw University of Technology  
Nowowiejska 15/19, 00-665 Warsaw, Poland  
e-mail: mkr@ii.pw.edu.pl

**Abstract.** Discovering association rules between items in a large database is an important database mining problem. The number of association rules may be huge. In this paper, we define a cover operator that logically derives a set of association rules from a given association rule. Representative association rules are defined as a least set of rules that covers all association rules satisfying certain user specified constraints. A user may be provided with a set of representative association rules instead of the whole set of association rules. The association rules, which are not representative ones, may be generated on demand by means of the cover operator. In this paper, we offer an algorithm computing representative association rules.

**Keywords:** representative association rules, cover operator,  $k$ -rule

## 1 Introduction

Discovering association rules between items in large databases is recognized as an important database mining problem. The problem was introduced in [1] for sales transaction database. The association rules identify sets of items that are purchased together with other sets of items. For example, an association rule may state that 90% of customers who buy butter and bread buy milk as well. Several extensions of the notion of an association rule were offered in the literature (see e.g. [2-4]). One of such extensions is generalized rules that can be discovered from taxonomic databases [2]. Applications for association rules range from decision support to telecommunications alarm diagnosis and prediction [5-6].

The number of association rules is usually huge. A user should not be presented with all of them, but rather with those which are original, novel, interesting. There were proposed several definitions of what is an interesting association rule (see e.g. [2, 7]). In particular, pruning out uninteresting rules which exploits the information in taxonomies seems to be quite useful (resulting in the rule number reduction amounting to 60% [2]). The interestingness of a rule is usually expressed by some quantitative measure. Our approach is somewhat different. We do not introduce any measure defining an interesting rule, but we show how to derive a set of association rules from a given association rule by means of a cover operator. A least set of association rules that allows to deduce all other rules satisfying certain user specified constraints will be called a set of representative association rules. A user could be provided with the set of representative association rules instead of the whole set of

association rules. The association rules, which are not representative ones, may be generated on demand by syntactic transformations of representative rules. In this paper, we propose an algorithm computing representative association rules.

## 2 Association Rules

The definition of a class of regularities, called *association rules*, and the problem of their discovering were introduced in [1]. Here, we describe this problem after [1,8]. Let  $I = \{i_1, i_2, \dots, i_m\}$  be a set of distinct literals, called *items*. In general, any set of items is called an *itemset*. Let  $\mathcal{D}$  be a set of transactions, where each transaction  $T$  is a set of items such that  $T \subseteq I$ . An *association rule* is an expression of the form  $X \Rightarrow Y$ , where  $\emptyset \neq X, Y \subset I$  and  $X \cap Y = \emptyset$ .  $X$  is called the antecedent and  $Y$  is called the consequent of the rule.

Statistical significance of an itemset  $X$  is called *support* and is denoted by  $sup(X)$ .  $Sup(X)$  is defined as the number of transactions in  $\mathcal{D}$  that contain  $X$ . Statistical significance (*support*) of a rule  $X \Rightarrow Y$  is denoted by  $sup(X \Rightarrow Y)$  and defined as  $sup(X \cup Y)$ . Additionally, an association rule is characterized by *confidence*, which expresses its strength. The confidence of an association rule  $X \Rightarrow Y$  is denoted by  $conf(X \Rightarrow Y)$  and defined as the ratio  $sup(X \cup Y) / sup(X)$ .

The problem of mining association rules is to generate all rules that have support greater than some user specified minimum support and confidence not less than user specified minimum confidence.

In the sequel, the set of all association rules whose support is greater than  $s$  and confidence is not less than  $c$  will be denoted by  $AR(s,c)$ . If  $s$  and  $c$  are understood then  $AR(s,c)$  will be denoted by  $AR$ .

In the paper, we apply also the following simple notions:

The number of items in an itemset will be called the *length of the itemset*. An itemset of the length  $k$  will be referred to as a  *$k$ -itemset*. Similarly, the *length of an association rule*  $X \Rightarrow Y$  will be defined as the total number of items in the rule's antecedent and consequent (i.e. in  $X \cup Y$ ). An association rule of the length  $k$  will be referred to as a  *$k$ -rule*. An association  $k$ -rule will be called *shorter* than, *longer* than or *of the same length* as an association  $m$ -rule if  $k < m$ ,  $k > m$ , or  $k = m$ , respectively.

## 3 Cover Operator

In this section, we introduce a notion of a *cover operator*, which will then be used to derive a set of association rules from a given association rule without accessing a database.

We define a *cover*  $C$  of the rule  $X \Rightarrow Y$ ,  $Y \neq \emptyset$ , as follows:

$$C(X \Rightarrow Y) = \{X \cup Z \Rightarrow V \mid Z, V \subseteq Y \text{ and } Z \cap V = \emptyset \text{ and } V \neq \emptyset\}.$$

Each rule in  $C(X \Rightarrow Y)$  consists of a subset of items occurring in the rule  $X \Rightarrow Y$ . The antecedent of any rule  $r$  covered by  $X \Rightarrow Y$  contains  $X$  and perhaps some items from  $Y$ , whereas  $r$ 's consequent is a non-empty subset of the remaining items in  $Y$ .

**Property 3.1**

Let  $r$  be an association rule having support  $s$  and confidence  $c$ .

Each rule  $r'$  belonging in the cover  $C(r)$  is an association rule whose support is not less than  $s$  and confidence is not less than  $c$ .

**Proof:**

Let us assume that  $r$  is the association rule of the form  $X \Rightarrow Y$  and  $r' \in C(r)$  and  $r'$  is the rule of the form  $(X \cup Z) \Rightarrow V$ . Let  $s'$  and  $c'$  denote support and confidence of  $r'$ , respectively. We are to prove that:

1.  $r'$  is an association rule,
2.  $s' \geq s$  and  $c' \geq c$ .

Ad. 1. Since  $r$  is an association rule then  $\emptyset \neq X, Y \subset I$  and  $X \cap Y = \emptyset$ . Additionally,  $Z, V \subseteq Y$  and  $Z \cap V = \emptyset$  and  $V \neq \emptyset$  because  $r' \in C(r)$ . Hence,  $X \cap V = \emptyset$  because  $X \cap Y = \emptyset$  and  $\emptyset \neq V \subseteq Y$ . Finally:

- $(X \cup Z) \neq \emptyset$  (since  $X \neq \emptyset$ ),
- $(X \cup Z) \subset I$  (since  $X \subset I$  and  $Z \subseteq Y \subset I$ ),
- $V \neq \emptyset$  (by definition of a rule belonging in the cover of an association rule),
- $V \subset I$  (since  $V \subseteq Y \subset I$ ),
- $(X \cup Z) \cap V = \emptyset$  (since  $Z \cap V = \emptyset$  and  $X \cap V = \emptyset$ ).

Ad. 2. Since  $r' \in C(r)$  then  $Z, V \subseteq Y$  and  $Z \cap V = \emptyset$ . Obviously,  $sup(X \cup Z) \leq sup(X)$ .

Additionally,  $sup(X \cup Z \cup V) \geq sup(X \cup Y)$  because  $Z \cup V \subseteq Y$ . Hence:

- $s' = sup(X \cup Z \cup V) \geq sup(X \cup Y) = s$ ,
- $c' = (s' / sup(X \cup Z)) \geq (s / sup(X)) = c$ .

□

The immediate important conclusion following from Property 3.1 is that if  $r$  belongs in  $AR(s, c)$  then every rule  $r'$  in  $C(r)$  also belongs in  $AR(s, c)$ . Hence, it is justified to consider  $C$  as an *operator of association inference*.

**Example 3.1**

Let  $T_1 = \{A, B, C, D, E\}$ ,  $T_2 = \{A, B, C, D, E, F\}$ ,  $T_3 = \{A, B, C, D, E, H, I\}$ ,  $T_4 = \{A, B, E\}$  and  $T_5 = \{B, C, D, E, H, I\}$  are the only transactions in the database  $\Delta$ . Let  $r: (AB \Rightarrow CDE)$ . Figure 1 contains all rules belonging in the cover  $C(r)$  along with their support and confidence in  $\Delta$ . The support of  $r$  is equal to 3 and its confidence is equal to 75%. The support and confidence of all other rules in  $C(r)$  are not less than the support and confidence of  $r$ .

□

**Property 3.2**

Let  $r: (X \Rightarrow Y)$  be an association rule. Then:

$$|C(r)| = 3^m - 2^m, \text{ where } m = |Y|.$$

| #   | Rule $r'$ in $C(r)$  | Support of $r'$ | Confidence of $r'$ |
|-----|----------------------|-----------------|--------------------|
| 1.  | $AB \Rightarrow CDE$ | 3               | 75%                |
| 2.  | $AB \Rightarrow CD$  | 3               | 75%                |
| 3.  | $AB \Rightarrow CE$  | 3               | 75%                |
| 4.  | $AB \Rightarrow DE$  | 3               | 75%                |
| 5.  | $AB \Rightarrow C$   | 3               | 75%                |
| 6.  | $AB \Rightarrow D$   | 3               | 75%                |
| 7.  | $AB \Rightarrow E$   | 4               | 100%               |
| 8.  | $ABC \Rightarrow DE$ | 3               | 100%               |
| 9.  | $ABC \Rightarrow D$  | 3               | 100%               |
| 19. | $ABC \Rightarrow E$  | 3               | 100%               |
| 11. | $ABD \Rightarrow CE$ | 3               | 100%               |
| 12. | $ABD \Rightarrow C$  | 3               | 100%               |
| 13. | $ABD \Rightarrow E$  | 3               | 100%               |
| 14. | $ABE \Rightarrow CD$ | 3               | 75%                |
| 15. | $ABE \Rightarrow C$  | 3               | 75%                |
| 16. | $ABE \Rightarrow D$  | 3               | 75%                |
| 17. | $ABCD \Rightarrow E$ | 3               | 100%               |
| 18. | $ABCE \Rightarrow D$ | 3               | 100%               |
| 19. | $ABDE \Rightarrow C$ | 3               | 100%               |

Fig. 1. The cover of the association rule  $r$ : ( $AB \Rightarrow CDE$ ) along with the rules' support and confidence.

### Proof:

Let  $F = \{X \cup Z \Rightarrow V \mid Z, V \subseteq Y \text{ and } Z \cap V = \emptyset\}$  and  $G = \{X \cup Z \Rightarrow \emptyset \mid Z \subseteq Y\}$ . It can be easily noticed that  $G \subset F$  and  $C(r) = F \setminus G$ . Hence,  $|C(r)| = |F| - |G|$ .

The rule set  $F$  can be obtained from the rule  $r$  by performing on each item  $y$  in  $Y$  one of the three basic operations:

1.  $y$  is removed from  $Y$ ,
2.  $y$  is moved to  $X$ ,
3.  $y$  remains in  $Y$ .

There are  $m$  items in  $Y$ , so the number of all possible different actions performed on the whole itemset  $Y$  is equal to  $3^m$ . One can easily observe that each action results in adding a new different rule to  $F$ . Thus,  $|F| = 3^m$ .

The rule set  $G$  can be obtained from the rule  $r$  by performing on each item  $y$  in  $Y$  one of the two basic operations:

1.  $y$  is removed from  $Y$ ,
2.  $y$  is moved to  $X$ .

Therefore,  $|G| = 2^m$ .

Finally,  $|C(r)| = |F| - |G| = 3^m - 2^m$ .

□

### Example 3.2

Let us consider the association rule  $r$ : ( $AB \Rightarrow CDE$ ) (see Example 3.1). Then:

$$|C(r)| = 3^3 - 2^3 = 27 - 8 = 19.$$

Thus,  $r$  represents 19 different association rules. □

### Property 3.3

Let  $r: (X \Rightarrow Y)$  and  $r': (X' \Rightarrow Y')$  be association rules. Then:

$$r \in C(r') \text{ iff } X \cup Y \subseteq X' \cup Y' \text{ and } X \supseteq X'.$$

#### Proof:

Property 3.3 follows immediately from the definition of the cover operator. □

### Property 3.4

- (i) If an association rule  $r$  is longer than an association rule  $r'$  then  $r \notin C(r')$ .
- (ii) If an association rule  $r: (X \Rightarrow Y)$  is shorter than an association rule  $r': (X' \Rightarrow Y')$  then

$$r \in C(r') \text{ iff } X \cup Y \subset X' \cup Y' \text{ and } X \supseteq X'.$$

- (i) If  $r: (X \Rightarrow Y)$  and  $r': (X' \Rightarrow Y')$  are different association rules of the same length then

$$r \in C(r') \text{ iff } X \cup Y = X' \cup Y' \text{ and } X \supsetneq X'.$$

#### Proof:

Property 3.4 follows immediately from Property 3.3. □

### Property 3.5 (of transitivity)

Let  $r, r'$  and  $r''$  be association rules.

$$\text{If } r \in C(r') \text{ and } r' \in C(r'') \text{ then } r \in C(r'').$$

#### Proof:

Let  $r: (X \Rightarrow Y)$ ,  $r': (X' \Rightarrow Y')$  and  $r'': (X'' \Rightarrow Y'')$ . Since  $r \in C(r')$  then, according to Property 3.3,  $X \cup Y \subseteq X' \cup Y'$  and  $X \supseteq X'$ . Similarly, since  $r' \in C(r'')$  then  $X' \cup Y' \subseteq X'' \cup Y''$  and  $X' \supseteq X''$ . Hence,  $X \cup Y \subseteq X'' \cup Y''$  and  $X \supseteq X''$ , which implies that  $r \in C(r'')$  (see Property 3.3). □

## 4 Representative Association Rules

In this section, we will introduce a notion of representative association rules. Informally speaking, a set of all representative association rules is a least set of rules that covers all association rules by means of the cover operator. We will investigate properties of representative rules which can be useful for the construction of a mining algorithm.

A set of *representative association rules* wrt. minimum support  $s$  and minimum confidence  $c$  will be denoted by  $RR(s,c)$  and defined as follows:

$$RR(s,c) = \{r \in AR(s,c) \mid \neg \exists r' \in AR(s,c), r' \neq r \text{ and } r \in C(r')\}.$$

If  $s$  and  $c$  are understood than  $RR(s,c)$  will be denoted by  $RR$ . Each rule in  $RR$  is called a *representative association rule*. By the definition of  $RR$ , no representative association rule may belong in the cover of another association rule.

#### Property 4.1

$$\text{If } r \in RR(s,c) \text{ then } C(r) \subseteq AR(s,c).$$

##### Proof:

By the definition of  $RR$ , if  $r \in RR(s,c)$  then  $r \in AR(s,c)$ . If  $r \in AR(s,c)$  then any rule in the cover  $C(r)$  belongs in  $AR(s,c)$  as well (see Property 3.1).  $\square$

#### Property 4.2

$$\forall r \in AR(s,c) \exists r' \in RR(s,c) \text{ such that } r \in C(r').$$

Let  $AR_k(s,c)$  be the set of all association rules in  $AR(s,c)$  of the length not less than  $k$ . Let  $RR_k(s,c)$  be the set of all representative association rules in  $RR(s,c)$  of the length not less than  $k$ .

#### Property 4.3

$$\forall r \in AR_k(s,c) \exists r' \in RR_k(s,c) \text{ such that } r \in C(r').$$

##### Proof:

Property 4.2 states that for each rule  $r \in AR_k(s,c)$  there is  $r'$  in  $RR(s,c)$  such that  $r \in C(r')$ . On the other hand, longer association rules do not belong in the covers of shorter association rules (see Property 3.4.i), so  $r$  must belong in the cover of some representative rule which is not shorter than  $r$ . Hence, for any rule  $r \in AR_k(s,c)$  there is  $r'$  in  $RR_k(s,c)$ , such that  $r \in C(r')$ .  $\square$

#### Property 4.4

Let  $r \in AR(s,c)$  be an association  $k$ -rule and  $l > k$ . Then:

$$\exists r' \in AR_l(s,c), r \in C(r') \text{ iff } \exists r'' \in RR_l(s,c), r \in C(r'').$$

##### Proof:

( $\Rightarrow$ )

Let us assume that there is  $r' \in AR_l(s,c)$  such that  $r \in C(r')$ . Since  $r' \in AR_l(s,c)$  then there is  $r''$  in  $RR_l(s,c)$  such that  $r' \in C(r'')$  (see Property 4.3). Since  $r \in C(r')$  and  $r' \in C(r'')$  then  $r \in C(r'')$  (see Property 3.5 of transitivity).

( $\Leftarrow$ )

Obvious.  $\square$

## 5 The Algorithm

The problem of generating association rules is usually decomposed into two subproblems:

1. Generate all itemsets whose support exceeds the minimum support  $minSup$ . The itemsets of this property are called *frequent (large)*.
2. Generate all association rules whose confidence is not less than the minimum confidence  $minConf$ . In the process of rules' generation only frequent itemsets are considered. Let  $X$  be a frequent itemset and  $\emptyset \neq Y \subset X$ . Then any rule  $X \setminus Y \Rightarrow Y$  holds if  $(sup(X) / sup(X \setminus Y)) \geq minConf$ .

We are interested in generating only representative association rules, so we will reformulate the second subproblem as follows:

- 2'. Generate all representative association rules whose confidence is not less than the minimum confidence.

Several efficient solutions were proposed to solve the first subproblem (see [2,8-10]). We will remind briefly the main idea of the *Apriori* algorithm [8] computing frequent itemsets. Then, we will propose an algorithm computing representative association rules from the found frequent itemsets.

### 5.1 Computing Frequent Itemsets

The *Apriori* algorithm exploits the following properties of frequent and non-frequent itemsets: All subsets of a frequent itemset are frequent and all supersets of a non-frequent itemset are non-frequent.

The following notation is used in the *Apriori* algorithm:

- $C_k$  - set of candidate  $k$ -itemsets;
- $F_k$  - set of frequent  $k$ -itemsets;

The items in itemsets are assumed to be ordered lexicographically. Associated with each itemset is a *count* field to store the support for this itemset.

#### Algorithm Apriori

```

 $F_1 = \{\text{frequent 1-itemsets}\};$
for ($k = 2; F_{k-1} \neq \emptyset; k++$) do begin
 $C_k = \text{AprioriGen}(F_{k-1});$
 forall transactions $t \in \mathcal{D}$ do
 forall candidates $c \in C_k$ do
 if $c \subseteq t$ then
 $c.\text{count}++;$
 $F_k = \{c \in C_k \mid c.\text{count} > minSup\};$
endfor;
return $\bigcup_k F_k;$
```

At the beginning the support of all 1-itemsets is determined during one pass over the database  $\mathcal{D}$ . All non-frequent 1-itemsets are discarded. Frequent 1-itemsets are used to generate candidate 2-itemsets by the *AprioriGen* function. Their support is

tested by scanning the database. Non-frequent 2-itemsets are discarded and new candidate 3-itemsets are computed by *AprioriGen* from frequent 2-itemsets. The procedure repeats in a similar way in the next iterations. In general, some  $k$ -th iteration consists of the following operations:

1. *AprioriGen* is called to generate the candidate  $k$ -itemsets  $C_k$  from the frequent  $(k-1)$ -itemsets  $F_{k-1}$ .
2. Supports for the candidate  $k$ -itemsets are determined by a pass over the database.
3. The candidate  $k$ -itemsets that do not exceed the minimum support are discarded; the remaining  $k$ -itemsets  $F_k$  are found frequent.

```
function AprioriGen(F_{k-1});
 insert into C_k
 select $f[1], f[2], \dots, f[k-1], c[k-1]$
 from $F_{k-1} f, F_{k-1} c$
 where $f[1] = c[1] \wedge \dots \wedge f[k-2] = c[k-2] \wedge f[k-1] < c[k-1];$
 delete all itemsets $c \in C_k$ such that some $(k-1)$ -subset of c is not in F_{k-1} ;
 return C_k ;
```

The *AprioriGen* function constructs candidate  $k$ -itemsets as supersets of frequent  $(k-1)$ -itemsets. This restriction of extending only frequent  $(k-1)$ -itemsets is justified since any  $k$ -itemset, which would be created as a result of extending a non-frequent  $(k-1)$ -itemset, would not be frequent either. The last operation in the function *AprioriGen* is pruning these candidates from  $C_k$  that do not have all their  $(k-1)$ -subsets in the frequent  $(k-1)$ -itemsets  $F_{k-1}$ . If  $k$ -itemset  $c$  does not have all its  $(k-1)$ -subsets in  $F_{k-1}$  then there is some non-frequent  $(k-1)$ -itemset  $c' \notin F_{k-1}$  which is a subset of  $c$ . This means that  $c$  is non-frequent as a superset of a non-frequent itemset.

Let us illustrate the *AprioriGen* function with an example. Let  $\{\{A,B\}, \{A,C\}, \{A,D\}, \{B,C\}, \{B,D\}\}$  be a family of frequent  $(k-1)$ -itemsets  $F_{k-1}$ . Then the subsequent candidate  $k$ -itemsets would be generated:  $\{A,B,C\}, \{A,B,D\}, \{A,C,D\}, \{B,C,D\}$ . Nevertheless, the final set of candidates is smaller. In the pruning process the candidates  $\{A,C,D\}$  and  $\{B,C,D\}$  will be discarded because there is not their subset  $\{C,D\}$  among  $(k-1)$ -itemsets in  $F_{k-1}$ .

## 5.2 Computing Representative Association Rules

The algorithm generating representative association rules, which we propose, is based on Properties 3.4 and 4.4 of association inference and the definition of *RR*. It generates association  $k$ -rules from  $k$ -itemsets,  $k=2, 3, \dots$ . It follows from Property 3.4.i that non/representativeness of a longer association rule does not depend on any shorter rule. On the other hand, Property 3.4.ii tells us that longer association rules may influence non/representativeness of shorter association rules. Additionally, Property 3.4.iii states that non/representativeness of any association  $k$ -rule generated from a  $k$ -itemset does not depend on association  $k$ -rules generated from other  $k$ -itemsets. Nevertheless, it is stated in Property 3.4.iii that non/representativeness of any association  $k$ -rule generated from a  $k$ -itemset may depend on other association  $k$ -rules generated from the same  $k$ -itemset. Property 3.4 allows us to conclude that among all association  $k$ -rules only association  $k$ -rules

whose antecedents are minimum itemsets are the representative ones unless they belong in the covers of longer association rules. According to Property 3.4.ii, an association rule  $X \Rightarrow Y$  belongs in the cover of a longer association rule  $X' \Rightarrow Y'$  if  $X \cup Y \subset X' \cup Y'$  and  $X \supseteq X'$ . The algorithm computing representative association rules applies also Property 4.4 to limit the number of longer rules tested. According to this property it is sufficient to confront the candidate rule  $r$  only with the representative association rules longer than  $r$  instead of confronting  $r$  with all association rules longer than  $r$ .

Notation used in the algorithm generating representative association rules:

- $F$  - set of frequent itemsets;
- $F_k$  - set of frequent  $k$ -itemsets;
- $RR$  - set of representative association rules wrt.  $\text{minSup}$  and  $\text{minConf}$ ,
- $R_k$  - set of representative association  $k$ -rules.

```
function GenAllRepresentatives(F);
 $RR = \emptyset$;
 $\text{maxK} =$ the cardinality of a set in F with maximal number of items;
 /* Loop1 */
 for $k = \text{maxK}$ to 2 do
 $RR = RR \cup \text{GenK-Representatives}(RR, F, k)$;
 return RR ;
```

The function *GenAllRepresentatives* controls the generation of representative association rules  $RR$  from the frequent itemsets (see Loop1). First, the longest (of the length  $\text{maxK}$ ) representative association rules  $RR$  are found by the *GenK-Representatives* function. Next, representative association rules shorter by 1 are found and added to  $RR$  etc. Finally, representative association 2-rules are generated by *GenK-Representatives* and included in  $RR$ . The computed set  $RR$  is the set of all representative association rules.

```
function GenK-Representatives(RR, F, k);
 $R_k = \emptyset$;
 forall $f \in F_k$ do begin
 $RR' = \{r \in RR \mid f \subset r.\text{antecedent} \cup r.\text{consequent}\}$;
 $A_1 = \{\{f[1]\}, \{f[2]\}, \dots, \{f[k]\}\}$; // candidate 1-antecedents
 /* Loop2 */
 for ($i = 1$; ($A_i \neq \emptyset$) and ($i < k$); $i++$) do begin
 forall $a \in A_i$ do begin
 find $f \in F$, such that $f = a$;
 $aCount = f.\text{count}$;
 if ($f.\text{count} / aCount \geq \text{minConf}$) and
 (there is no $r \in RR'$ such that $a \supseteq r.\text{antecedent}$) then begin
 $r.\text{antecedent} = a$;
 $r.\text{consequent} = f \setminus a$;
 $r.\text{support} = f.\text{count}$;
 $r.\text{confidence} = f.\text{count} / aCount$;
```

```

 $R_k = R_k \cup \{r\};$
 $A_i = A_i \setminus \{a\};$
endif;
endfor;
 $A_{i+1} = \text{AprioriGen}(A_i); \quad // \text{compute candidate } i+1\text{-antecedents}$
endfor;
endfor;
return $R_k;$

```

The *GenK-Representatives* function has three parameters:

- the parameter  $k$  determines the length of representative association rules to be computed,
- $RR$  - is the set of representative association rules found so far; all rules in  $RR$  are longer than  $k$ ,
- $F$  - is the set of all frequent itemsets.

*GenK-Representatives* computes representative association  $k$ -rules from each itemset in  $F_k$ . Let  $f$  be a considered itemset in  $F_k$ . The representative association  $k$ -rules are generated from  $f$  as follows:

First two sets  $RR'$  and  $A_1$  are created.  $RR'$  is the set of representative association rules longer than  $k$  which contain all items occurring in  $f$ .  $A_1$  is assigned the family of singletons corresponding to items in  $f$ . Each itemset  $a$  in  $A_1$  is treated as an antecedent of a candidate  $k$ -rule  $a \Rightarrow f \setminus a$ . First iteration of Loop 2 starts. The confidence of each candidate rule is computed in order to check whether it is an association rule. The support of the rule is equal to  $f.count$ , while the support of the antecedent  $a$  is found as the support of the respective itemset  $f$  in  $F_1$ . If the rule's confidence is not less than  $\minConf$  then representativeness of the rule is tested. The test on the rule's representativeness is performed in accordance with Property 3.4.ii and consists in confronting  $a \Rightarrow f \setminus a$  with all representative association rules in  $RR'$ . If no rule in  $RR'$  covers the rule  $a \Rightarrow f \setminus a$  then  $a \Rightarrow f \setminus a$  is placed in  $R_k$  and  $a$  is removed from the set  $A_1$ . After having tested all candidate rules,  $A_1$  may still be non-empty - the remaining itemsets turned out not to be antecedents of representative association rules. The remaining itemsets in  $A_1$  will be used by the *AprioriGen* function to generate  $A_2$ , which will contain 2-item antecedents of new candidate rules. *AprioriGen* guarantees that no itemset in  $A_2$  is a superset of an antecedent of any representative association rule placed in  $R_k$  so far. This means that no rule  $a \Rightarrow f \setminus a$ , where  $a \in A_2$ , belongs in the cover of any rule in  $R_k$ . After  $A_2$  has been determined the second iteration of Loop 2 starts. All iterations of Loop 2 are performed in the same way as the first iteration. Each subsequent iteration  $i$  ends in determination of antecedents  $A_{i+1}$  of new candidate rules. Loop 2 is performed as long as newly determined  $A_i$  is empty.

The described procedure is repeated for each  $k$ -itemset  $f$  in  $F_k$ . Finally, the *GenK-Representatives* function returns the set of all representative association  $k$ -rules  $R_k$ .

**Example 5.1**

Given  $\text{minSup} = 3$  and  $\text{minConf} = 75\%$ , the following representative rules would be found for the database  $\Delta$  from Example 3.1:

$$\begin{aligned} RR(3, 75\%) = \{ & A \Rightarrow BCDE, C \Rightarrow ABDE, D \Rightarrow ABCE, B \Rightarrow CDE, \\ & E \Rightarrow BCD, B \Rightarrow AE, E \Rightarrow AB \}. \end{aligned}$$

There are 7 representative association rules in  $RR(3, 75\%)$ , whereas the number of all association rules in  $AR(3, 75\%)$  is 165. Hence, the representative association rules constitute 4.24% of all association rules.

□

## 6 Mining Around Representative Association Rules

A user may request to be provided with the set of all association rules  $AR$  or the set of representative association rules  $RR$ . If  $RR$  is discovered then the user may formulate queries about the association rules represented by  $RR$ . The queries may contain not only the cover operator  $C$ , but also the set-theoretical operators of union, difference and intersection.

**Example 6.1**

Let us assume that a user first looks for the representative rules  $RR$ . Let  $r, r' \in RR$  be rules interesting for him. The user decides to see all rules represented by  $r$ . To this end he issues a simple query:  $C(r)$ . Next, he wants to see the rules represented by the rule  $r'$  that were not listed yet. He may specify his demand as follows:  $C(r') \setminus C(r)$ . If the user is interested in the set of rules represented both by  $r$  and  $r'$ , he will ask:  $C(r') \cap C(r)$  etc.

□

## 7 Conclusions

In the paper we introduced a notion of a cover operator which transforms an association rule into the set of association rules by syntactic transformations of the initial rule. The representative association rules are defined as a least set of rules that represents all association rules satisfying certain user specified support and confidence. A user may be provided with the set of representative association rules instead of the whole, usually huge, set of association rules. The association rules, which are not representative ones, may be generated on demand by means of the cover operator. The algorithm generating representative association rules was presented in the paper.

## Acknowledgements

I would like to thank Prof. Henryk Rybiński for his comments on this paper.

## References

1. Agraval R., Imielinski T., Swami A., Mining Associations Rules between Sets of Items in Large Databases. In *Proc. of the ACM SIGMOD Conference on Management of Data*, Washington, D.C., May 1993, pp. 207-216.
2. Srikant R., Agraval R., Mining Generalized Association Rules. In *Proc. of the 21st VLDB Conference*, Zurich, Switzerland, 1995, pp. 407-419.
3. Imielinski T., Virmani A., Abdulghani A., Discover Board Application Programming Interface and Query Language for Database Mining. In *Proc. of KDD '96*, Portland Ore., August 1996, pp. 20-26.
4. Meo R., Psaila G., Ceri S., A New SQL-like Operator for Mining Association Rules, Proc. of the 22nd VLDB Conference, Mumbai (Bombay), India, 1996.
5. *Communications of the ACM*, November 1996 - Vol. 39, No 11.
6. Advances in Knowledge Discovery and Data Mining, eds. U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, R. Uthurusamy, AAAI, Menlo Park, California, 1996.
7. Piatetsky-Shapiro G., Discovery, Analysis and Presentation of Strong Rules. In *Knowledge Discovery in Databases*, G. Piatetsky-Shapiro, W. Frawley, eds., AAAI/MIT Press, Menlo Park, CA, 1991, pp. 229-248.
8. Agraval R., Mannila H., Srikant R., Toivonen H., Verkamo A.I., Fast Discovery of Association Rules. In *Advances in Knowledge Discovery and Data Mining*, U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, R. Uthurusamy, eds., AAAI, Menlo Park, California, 1996, pp. 307-328.
9. Savasere A., Omiecinski E., Navathe S., An Efficient Algorithm for Mining Association Rules in Large Databases. In *Proc. of the 21st VLDB Conference*, Zurich, Switzerland, 1995, pp. 432-444.
10. Houtsma M., Swami A., Set-oriented Mining of Association Rules. In Int'l Conference on Data Engineering, Taipei, Taiwan, March 1995.

# Identifying Relevant Databases for Multidatabase Mining

Huan Liu, Hongjun Lu, Jun Yao

Department of Information Systems and Computer Science  
National University of Singapore  
Kent Ridge, Singapore 119260  
[{liuh,luhj,yaojun}@iscs.nus.edu.sg](mailto:{liuh,luhj,yaojun}@iscs.nus.edu.sg)

**Abstract.** Various tools and systems for knowledge discovery and data mining are developed and available for applications. However, when we are immersed in heaps of databases, an immediate question facing practitioners is where we should start mining. In this paper, breaking away from the conventional data mining assumption that many databases be joined into one, we argue that the first step for multidatabase mining is to identify databases that are most likely relevant to an application; without doing so, the mining process can be lengthy, aimless and ineffective. A relevance measure is thus proposed to identify relevant databases for mining tasks with an objective to find patterns or regularities about certain attributes. An efficient implementation for identifying relevant databases is described. Experiments are conducted to validate the measure's performance and to show its promising applications.

**Keywords:** multiple databases, data mining, query, relevance measure

## 1 Introduction

While researchers are trying to develop efficient algorithms to cope with a large volume of data [1, 5, 18], little work has been devoted to the *data aspect* in knowledge discovery process. Although most data mining algorithms assume a single data set, for real world application practitioners have to face the problem of discovering knowledge from systems with multiple databases. To mine knowledge from multiple databases, a brute force approach is to join the related tables into a single large table, upon which existing data mining techniques or tools can be applied. There are several problems for this approach in real world application. First, database integration itself is still a problematic area, especially where the source domains differ. Second, the size of the joined table will increase. The increase in size not only prolongs the running time of mining algorithms, but also affects the behavior of mining algorithms. Third, among a large number of generated patterns in a database, it is often difficult for users to search for useful ones [9, 13, 16]. The useless or uninteresting patterns are generated as there are quite a number of irrelevant databases when joining. Therefore, as in any effective knowledge discovery process, the first important step in mining

from multiple databases is indeed to select those databases that are *relevant* to the specific mining task.

In this paper, we will address the relevance problem: identifying databases relevant to a particular data mining task in multiple databases. Without the loss of generality, we call each database a relation, or a table and assume that (1) a specific mining task is related to the property of certain attributes, which can be expressed using query predicates and (2) the higher order correlations of attributes with a query are at least partially reflected in their first order correlations. The problem of identifying relevant databases can be stated as follows:

Given  $n$  data tables (relations),  $D_k (1 \leq k \leq n)$ , each of which consists of a number of attributes. A query predicate  $Q$  is expressed in the form of " $A_i \text{ relop } C$ " where  $A_i$  is an attribute, *relop* is one of the relational operators ( $=, >, <, \leq, \geq, \neq$ ), and  $C$  is a value in its domain. Identifying relevant databases is to select *relevant* tables (relations) that contain specific, reliable and statistically significant information pertaining to the query.

We start with mining tasks with query predicates for two reasons. (1) Queries are easy to form and are most familiar and natural when dealing with databases: user's interest, prior knowledge and intention can be conveyed by means of such queries; (2) among various data mining tasks (classification, characterization, generalization, association, etc.), most of them aim at discovering knowledge that can be expressed as relationships among values of attributes [3, 11, 6, 15, 17]. What we are concerned is to have a relatively simple and fast method by which we can identify the databases (tables) that are relevant to a mining task. After that, various existing techniques can be used to conduct mining tasks. Major contributions of our work include:

- illustrating the importance and necessity of identifying relevant databases when mining from multi-databases;
- establishing a quantitative measure that allows us to identify relevant database (tables) before mining; and
- developing an efficient implementation of computing the relevance measure.

The remainder of the paper is organized as follows. Section 2 gives a motivating example to further illustrate the issue of relevance. Section 3 presents a relevance measure to identify relevant databases. Section 4 describes an algorithm that identifies relevant databases using the proposed measure. Section 5 presents some preliminary results of using the measure to a real database. Section 6 discusses the relationship between our work and some related work. Section 7 concludes the paper.

## 2 A Motivating Example

In this section, we will use two artificial databases to argue that, for data mining tasks involving multiple databases, it is better to identify relevant database first before applying mining techniques.

Our example databases are about people's diet habits. It is user's intention to discover some *useful* knowledge about Chinese food habits from the databases. Here and in most KDD applications, being useful means something that should potentially lead to some useful actions by a user [4]. Specific information is one type of usefulness. In our example, the useful knowledge should be some diet habits specific to Chinese. Two tables (databases) with desired attributes, are as shown in Tables 1 and 2.

**Table 1.** Database 1 of Diet Habits

| ID     | ethnic-group | alcohol   | on-diet | snack-between-meals | favorite-non-veg |
|--------|--------------|-----------|---------|---------------------|------------------|
| 950351 | Chinese      | never     | no      | sometimes           | fish             |
| 950301 | Chinese      | never     | no      | seldom              | pork             |
| 950282 | Chinese      | sometimes | no      | seldom              | fish             |
| 940112 | Chinese      | often     | no      | often               | chicken          |
| 940023 | Chinese      | sometimes | no      | sometimes           | beef             |
| 938976 | Chinese      | sometimes | no      | sometimes           | egg              |
| 950612 | American     | never     | no      | seldom              | beef             |
| 950122 | American     | often     | no      | sometimes           | beef             |
| 940227 | American     | sometimes | no      | sometimes           | chicken          |
| 938567 | American     | sometimes | no      | often               | pork             |
| 950348 | Indian       | often     | no      | sometimes           | fish             |
| 950312 | Indian       | sometimes | no      | seldom              | pork             |
| 950123 | Indian       | never     | no      | sometimes           | chicken          |
| 940247 | Indian       | sometimes | no      | sometimes           | fish             |
| 940100 | Indian       | never     | no      | sometimes           | beef             |

Let's look at Table 1 first. It is difficult to draw a conclusion that the database contains information specific to Chinese. In such a case, it is fair to say that the database is irrelevant with respect to the query about Chinese.

It is easy to see that database in Table 2 is relevant to the query about Chinese since we can at least derive such a statement (rule)

*"If ethnic group is Chinese, the main food is rice"*

since all records about Chinese show that the main food is rice whereas records about American and Indian do not indicate such a habit. We have seen that although both databases contain information about the diet habits of Chinese, one of them contains relevant knowledge but the other does not.

We were able to manually pick one database over the other in the above simple example because the databases are small. We need a method for large databases for selection. Let's examine the existing method to see if some of them can serve the purpose of data mining, pretending no knowledge about which database contains interesting information.

**Table 2.** Database 2 of Diet Habits

| ID     | ethnic-Group | main-food | regular eating times | drink  | vegetarian |
|--------|--------------|-----------|----------------------|--------|------------|
| 950578 | Chinese      | rice      | 3                    | tea    | no         |
| 950351 | Chinese      | rice      | 3                    | tea    | no         |
| 950301 | Chinese      | rice      | 3                    | tea    | no         |
| 950282 | Chinese      | rice      | 3                    | tea    | no         |
| 940226 | Chinese      | rice      | 3                    | cola   | no         |
| 940112 | Chinese      | rice      | 3                    | tea    | yes        |
| 950612 | American     | bread     | 3                    | coffee | no         |
| 950122 | American     | bread     | 3                    | cola   | no         |
| 940227 | American     | rice      | 3                    | cola   | yes        |
| 940121 | American     | bread     | 3                    | tea    | no         |
| 950348 | Indian       | bread     | 3                    | coffee | no         |
| 950312 | Indian       | bread     | 3                    | coffee | yes        |
| 950123 | Indian       | rice      | 3                    | cola   | no         |
| 940247 | Indian       | rice      | 3                    | coffee | no         |
| 940109 | Indian       | bread     | 3                    | tea    | no         |

1. The first solution is to combine two tables together, i.e., joining them on the common attribute *ID* before applying mining algorithms. There are some problems of doing so. (1) the resulting tuples will have 10 attributes besides *ID*, i.e., as twice as many in the original databases; (2) the resulting table will have more tuples. In our example, the number of samples increases to 19 (4 more than the original one); and (3) for certain samples, some attributes will have missing values since they only appear in one of the databases. All these will surely make the subsequent mining task more complex.
2. The second solution could be issuing database queries to retrieve information from the databases and see which one should be focused on. In our example, if an SQL selection query with condition “**WHERE** ethnic-group = Chinese” is posed, both databases will return 6 tuples. There is no indication whether one of them is relevant or not.
3. We can apply some data mining algorithms to the two databases, some patterns will be generated from the irrelevant database. For instance, using the concept of *support* and *confidence* level in mining association rules, we can have rules like:

$$\text{ethnic-group} = \text{Chinese} \rightarrow \text{on-diet} = \text{no}$$

with 40% support and the confidence level of 100%. We do not think such information is really interesting since *on-diet* = no in fact holds for all tuples. We can also apply certain classification algorithms such as CART, CN2, and C4.5 [2, 3, 15] to the database specifying attribute *ethnic-group* as the class

label. We will get some classification rules with certain accuracy too but not interesting.

From the above observations, several possible solutions seem inadequate to efficiently distinguish the irrelevant database from the relevant one. It is clear that there is a necessity of identifying relevant databases first when mining from multiple data sources.

### 3 Relevance Measurement

In this section, we discuss a quantitative measure of relevance. The relevant databases identified by the measures should contain specific information pertaining to our mining task, which is reliable and statistically significant.

We call  $A$  *relop*  $C$  a *selector*, where  $A$  is an attribute name which is not referenced by the query predicate  $Q$ ,  $\text{relop} \in \{=, <, >, \leq, \geq, \neq\}$ , and  $C$  is a constant value in the domain of  $A$ . We define the *relevance factor* of selector  $s$  with respect to  $Q$  as:

$$RF(s, Q) = \Pr(s|Q) \Pr(Q) \log \frac{\Pr(s|Q)}{\Pr(s)} \quad (1)$$

where  $\Pr(Q)$  and  $\Pr(s)$  are priors and estimated by the ratios how frequently they appear in a database; and  $\Pr(s|Q)$  is the posterior about the frequency ratio of  $s$  appearing given that  $Q$  occurs. The rationale of defining relevance as in Equation 1 is as follows:

$\frac{\Pr(s|Q)}{\Pr(s)}$  shows degree of the deviation of the posterior from the prior. This ratio tells us different relationships between  $\Pr(s|Q)$  and  $\Pr(s)$ .

**Case 1** If  $\frac{\Pr(s|Q)}{\Pr(s)}$  is close to 1, i.e.,  $\Pr(s|Q) \simeq \Pr(s)$ ,  $s$  is independent of  $Q$ ;

**Case 2** If  $\frac{\Pr(s|Q)}{\Pr(s)}$  is close to 0, i.e.,  $\Pr(s|Q)$  is almost 0,  $s$  rarely occurs given  $Q$ ;

**Case 3** If  $\frac{\Pr(s|Q)}{\Pr(s)}$  is less than 1,  $s$  is not frequent enough using  $\Pr(Q)$  as reference;

**Case 4** If  $\frac{\Pr(s|Q)}{\Pr(s)}$  is greater than 1, then  $s$  occurs more often given  $Q$  than without  $Q$ , hence  $s$  and  $Q$  are correlated.

It is clear that in the context of finding relevant databases, we are only interested in the last case. Therefore, the logarithm function is taken since  $\log \frac{\Pr(s|Q)}{\Pr(s)}$  is either 0 or less than 0 for cases 1 and 3. Adding a weight  $\Pr(s|Q)$  to the measure, we make sure that in case 2,  $RF$  is also close to 0.  $\Pr(s|Q)$  also indicates how valid the correlation between  $s$  and  $Q$  is. Another weight  $\Pr(Q)$  is introduced to take into account how frequently  $Q$  occurs in the database. This is because

**Case 5** If  $\Pr(Q)$  is close to 0,  $Q$  seldom occurs in the database, hence it may not be statistically significant to conclude something related to  $Q$ .

Thus, we develop a measure  $RF$  that is able to distinguish all the five cases. The higher the value  $RF$ , the more relevant a selector  $s$  with respect to query  $Q$ . If  $RF$  for all selectors of a database is close to or less than 0, the database is irrelevant to  $Q$ .

With the above definition about the relevance factor of selectors, we can have definition about the relevance of databases.

A selector  $s$  is relevant to  $Q$  if  $RF(s, Q) > \delta$ , where  $\delta (> 0)$  is the given threshold.

A table is relevant to  $Q$  if there exists at least one selector  $s_j$ , ( $A_i \text{ relop } C$ ) where  $s_j$  is relevant to  $Q$ .

Now we have a quantitative approach to distinguishing irrelevant databases from the relevant ones. Let us compute the values of  $RF$  for the example databases in the previous section. Given  $Q: \text{Ethnic-Group} = \text{Chinese}$ . The maximum  $RF$  in Table 1 value is 0.088. If we set  $\delta$  to 0.1, the database will be considered as irrelevant one since there does not exist any selector relevant to  $Q$ . With the same value of  $\delta$ , two relevant selectors in Table 2, *main food = rice* and *drink = tea* with  $RF$  values as 0.294 and 0.278 respectively, are considered relevant to  $Q$ . We conclude that the database is relevant. From the example of these tables, we can see that the definition can tell relevant database from irrelevant one.

One issue in using the relevance measure is how to choose the threshold  $\delta$ . The definition of  $RF$  consists of two parts:  $\Pr(s|Q) \times \Pr(Q)$  - the term before the logarithm and  $\Pr(s|Q)/\Pr(s)$  - the logarithmic term. In our definition, the selectors with larger values for both parts will be considered more specific to a query. In Figure 1, we draw  $\Pr(Q \wedge s) \times \log \Pr(s|Q)/\Pr(s) = \delta_i$  for  $\delta_i = 0.05, 0.1, 0.2$  and 0.5 using  $\Pr(Q \wedge s)$  and  $\Pr(s|Q)/\Pr(s)$  as variables for the y-axis and x-axis respectively. For a threshold value  $\delta_i$ , a database is identified as relevant one only if it contains at least one selector  $s$  such that  $\Pr(Q \wedge s)$  and  $\Pr(s|Q)/\Pr(s)$  fall into the right-upper region of the curve in Figure 1. However,  $\Pr(Q \wedge s)$  is usually not very high in real databases, especially in large databases. The value of  $RF$  will decrease rapidly as the value of  $\Pr(Q \wedge s)$  decreases. So we need to lower the threshold when database size is large.

For most data mining tasks,  $\Pr(Q \wedge s)$ , i.e., the percentage of samples for which both  $Q$  and  $s$  hold should not be too low, since this expectation value indicates the relative significance in terms of statistics. But it is also not obvious what lower bound of  $\Pr(Q \wedge s)$  is. In order to make sure that  $\Pr(Q \wedge s)$  be not too small, the query predicate should cover a sufficient number of tuples in the database at first place, i.e.,  $\Pr(Q)$  should not be too small. Usually, we know the total number of samples in a database. Therefore it is relatively easy to specify a lower bound of  $\Pr(Q)$  (the percentage of samples covered by a query). We can proof that, given  $\Pr(Q)$  the following holds:  $RF \leq \Pr(Q) \log(1/\Pr(Q))$

Therefore, given an estimate of  $\Pr(Q)$ , we can find the maximum value of  $RF$  from Figure 2. To identify a relevant database, we can choose  $\delta$  depending on our expectation of the degree of relevance. With a small  $\delta$ , more databases will

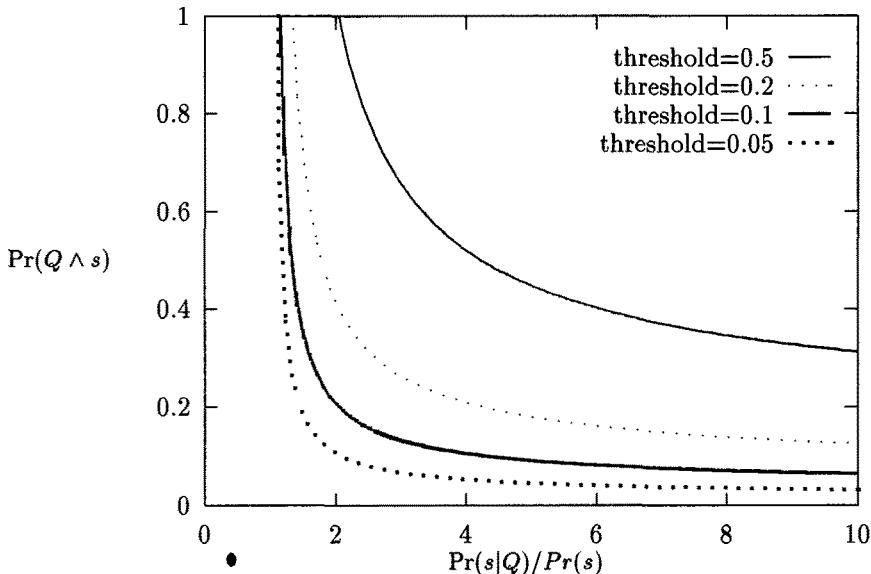


Fig. 1.  $\text{Pr}(Q \wedge s)$  as the function of  $\text{Pr}(s|Q)/\text{Pr}(s)$  on different threshold values

be considered as relevant. It might be reasonable to set threshold value  $\delta$  about one-third of the maximum value of  $RF$ . In the previous example,  $Q$  is *ethnic-group = Chinese* and  $\text{Pr}(Q)$  is 40%. The corresponding maximum value of  $RF$  is about 0.50. If we choose 0.16 as the threshold value for both  $\delta$ ,  $RF$  values in the second database all are greater than 0.16 and identified as the relevant one. If we check the values of  $RF$  in the first database, neither of them is greater than 0.16. Therefore it is irrelevant.

#### 4 Identifying Relevant Databases

With the relevance factor  $RF$ , we can determine whether a database is relevant to a query predicate before applying data mining algorithms. To compute  $RF(s, Q)$ , we only need to count three values,  $\text{Pr}(Q)$ ,  $\text{Pr}(s)$  and  $\text{Pr}(s \wedge Q)$ . To determine whether a database is relevant to  $Q$ , we need to test all selectors, which can be done by scanning the table once. Let us assume that there are  $m + 1$  attributes,  $A_0, \dots, A_m$ . Attribute  $A_0$  is referenced by  $Q$ . For each of the other attributes,  $A_i$ ,  $1 \leq i \leq m$ , a table  $S_i$  is maintained to keep track of selectors and the related counters. An entry of  $S_i$  is a triple of  $(S\_value, S\_counter, SQ\_counter)$ .  $S\_value$  is the value of selector,  $S\_counter$

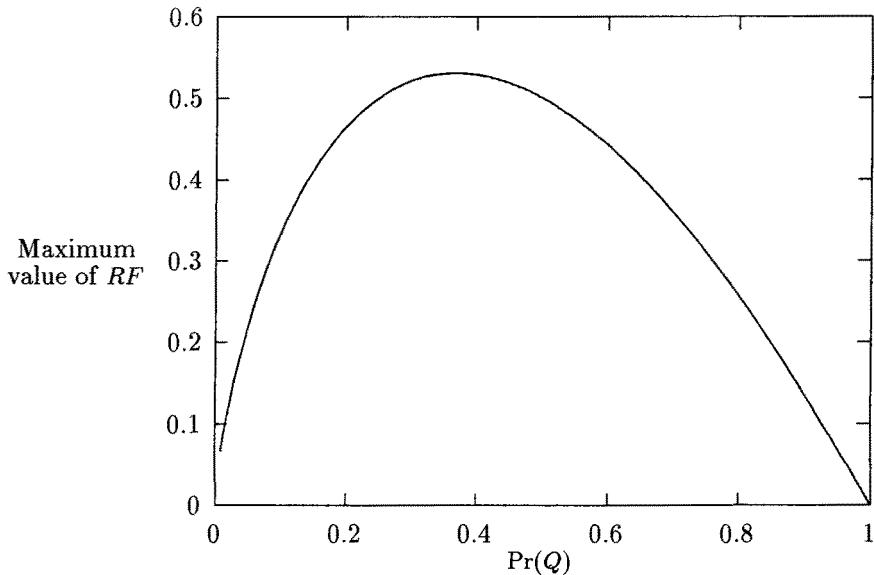


Fig. 2. Maximum value of  $RF$  as the function of  $\text{Pr}(Q)$

records the number of tuples for which  $A_i = S\_value$  is true.  $SQ\_counter$  records the number of tuples for which both  $Q$  and  $A_i = S\_value$  are true. With  $S_i$  table, we can determine the relevance of a database by scanning it once. This process has two parts: (1) reading each record in the database; and (2) searching for the entry of selectors and update the counters for each attribute in the record. The cost of part 1 is proportional to the number of records in the database. As for part 2, with a proper data structure, for example, using hashing function for the selector tables, the search for selector entries can be kept as constant, regardless of the number of selectors of an attribute. Therefore, the run time for the overall calculation is  $O(NM)$  where  $N$  is the number of records in the database and  $M$  is the number of attributes.

## 5 Experiments

In order to have a better understanding of the issue of relevance and the relevance factor, we conducted some preliminary experiments on real-world databases using the relevance measure. To simplify the task of finding meaningful databases, we transform the problem of identifying relevant databases from multiple databases to a problem of determining the relevance of a single database to different query

predicates. One of the databases tested is the NSERC (Natural Science and Engineering Research Council of Canada) research grant database. The database contains 8 tables. We take the largest table, NSERC table, as the example. The table contains information on amount of awards with 14 attributes as follows: *Id*, *Sysid*, *Sortname*, *Dept*, *Organization-id*, *Fyr*, *Compyr*, *Award*, *Grand-code*, *Ctee*, *Install*, *Acd3*, *Discipline-code* and *Cnt*.

One of our objectives is to determine whether the table contains interesting information related to the grant amount *Award*. First we need to determine the threshold. We estimated the maximum *RF* as follows: The total number of tuples in the table is 18510. To make a query predicate have statistical significance, we require that the query covers at least 3% of the tuples (about 500). So we set the lower bound for  $\Pr(Q)$  as 0.03. From Figure 2 we can find that the maximum *RF* is about 0.15. Therefore we set the threshold value as 0.05.

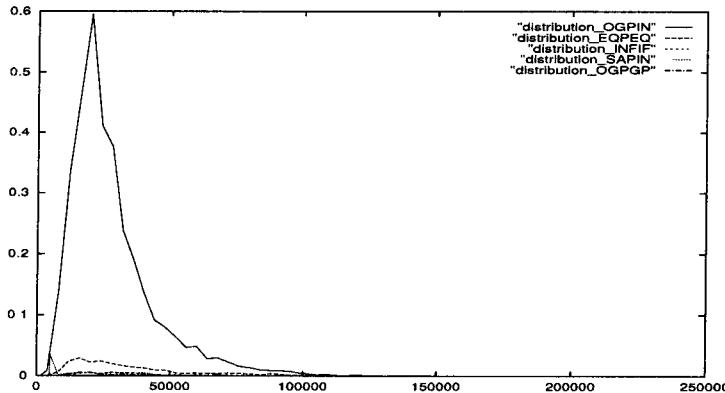
**Table 3.** Experiments with NSERC data

| Query predicate $Q$        | Selector $S$ where $RF(s, Q) > \delta$ | $RF(s, Q)$ |
|----------------------------|----------------------------------------|------------|
| $0 \leq Award < 20000$     | <i>Grand-code</i> = OGPIN              | 0.096328   |
|                            | <i>Sysid</i> = G                       | 0.088566   |
|                            | <i>Acd3</i> = 1201                     | 0.060338   |
| $20000 \leq Award < 40000$ | <i>Grand-code</i> = OGPIN              | 0.091384   |
|                            | <i>Sysid</i> = G                       | 0.091384   |
| $40000 \leq Award < 60000$ | Nil                                    |            |
| $60000 \leq Award < 90000$ | Nil                                    |            |
| $90000 \leq Award$         | Nil                                    |            |

Since attribute *Award*, the amount of award is continuous, we construct query predicates in the form of:  $a_1 \leq Award < a_2$ . Table 3 lists the testing results of five query predicates. We can see that, for two queries, with the award amount between [0, 20000) and [20000, 40000), we obtain some selectors whose *RF* values are greater than the specified threshold 0.05. For other three query predicates, no such selector is obtained. In other words, NSERC table is relevant to the first two queries, but irrelevant to the last three queries. That is, we are able to find interesting patterns about award amount in the first two ranges but not in the last three.

To verify this, let us take a close look at selector *Grand-code* = OGPIN (OGPIN refers to individual research grant type), since the *RF* value of *Grand-code* = OGPIN for both query predicates are greater than the threshold. Figure 5 shows the distributions of amount of award for top five grant codes: OGPIN, EQPEQ, INFIF, SAPIN, and OGPGP. The area between a curve for a grant code and the x-axis is the number of tuples with that code. As we can see, most tuples with grant code OGPIN have the amount of Award in the range of

(0, 40000), that is the distribution of award amount for OGPIN has a speciality within this range, therefore the table is identified as relevant to the queries in these two ranges. Viewing these distributions and their shapes can also determine which code is relevant to the query. However, it would involve human interpretation. It is not practical when the number of codes is large and with many databases to examine. Our measure implements human interpretation in an automatic manner.



**Fig. 3.** Distributions of amount of Award

## 6 Related Work

To our knowledge, little work on mining from multidatabase has been reported in the literature. However, research work on interestingness of discovered knowledge seems quite related to this work, though the objectives and methods are different.

As mentioned before, applying data mining techniques to a given database can usually generate some knowledge in the forms such as patterns or rules. Typically, the number of such patterns is large and only a small fraction of which may be of interest to the users. Research work on interestingness of knowledge tried to distinguish the potentially interesting patterns from others. Some researchers proposed approaches that determine interestingness of a discovered rule based on user's feedback [16, 10, 14, 8], hence the measurement of interestingness is *subjective*. Researchers also developed various *objective* interestingness measures based on the statistics underlying the discovered patterns [12, 7]. With the quantitative objective interestingness measure, discovered patterns can be ranked and less interesting ones can be filtered out. Kamber and Shinghal proposed a measure,  $IC^{++}$ , which is claimed to have the best property regarding evaluating the interestingness of characteristic rules.

While interestingness is an important issue for knowledge discovery, we argue that it is also important to determine whether a database contains potentially interesting information before applying data mining techniques. When we filter out those irrelevant databases, we should not discard the databases that may contain interesting information. To check on this, we apply our method to the example given by [7]. The results show that the relevance measure serves the purpose of identifying relevant databases: For the database relevant to the query, interesting rules can be found. Otherwise, the discovered rules are not interesting. Notice that mining rules is more costly than identifying relevant databases. It is quite reasonable to identifying the relevant databases at first before mining algorithms applied.

## 7 Conclusions

In this paper, we addressed one of the issues related to data mining from multi-database systems: identifying those databases relevant to a data mining task from a set of semantically related databases. We argue that effective data mining from multi-databases should involve a preselection phase. Data mining techniques are only applied to those relevant databases. In order to quantify the relevance of a database, a measurement,  $RF$ , relevance factor, is proposed. The beauty of  $RF$  is that, (1) its value can indicate statistical significance of different aspects of an attribute value related to the query predicate; and (2) it can be efficiently computed by scanning the database only once.

Some preliminary experiments were conducted to test the method proposed in the paper and the measure was cross validated by the state-of-art interesting measure in the literature. One future direction of research is providing a better guideline for choosing the threshold value since sometimes it may be not easy to determine a proper threshold value. Also, we look forward to more comprehensive experiments on real world data.

### Acknowledgement

We like to thank Jiawei Han who makes the NSERC database available to us.

## References

1. R. Agrawal, T. Imielinski, and A. Swami. Database mining: A performance perspective. *IEEE Transactions on Knowledge and Data Engineering*, 5(6):914–925, Dec 1993.
2. L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression*. Wadsworth & Brooks/Cole Advanced & Books Software, 1984.
3. P. Clark and T. Niblett. The CN2 induction algorithm. *Machine Learning*, 3:261–283, 1989.
4. U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery: An overview. In U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 1–34. AAAI Press / The MIT Press, 1996.

5. J. Han and Y. Fu. Attribute-oriented induction in data mining. In U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 399–421. AAAI Press / The MIT Press, 1996.
6. J. Hong and C. Mao. Incremental discovery of rules and structure by hierarchical and parallel clustering. In G. Piatetsky-Shapiro and W. J. Frawley, editors, *Knowledge Discovery in Databases*, pages 177–194. AAAI / The MIT Press, 1991.
7. M. Kamber and R. Shinghal. Evaluating the interestingness of characteristic rules. In *Proceedings of the Second International Conference on Data Mining (KDD-96)*, pages 263–266. AAAI Press, 1996.
8. B. Liu and W. Hsu. Post-analysis of learned rules. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence AAAI-96*, pages 828–834. AAAI press / The MIT press, August 1996.
9. J. A. Major and J. Mangano. Selecting among rules induced from a hurricane database. In G. Piatetsky-Shapiro, editor, *Proceedings of AAAI-93 workshop on Knowledge Discovery in Databases*, pages 28–44, 1993.
10. C. J. Matheus, G. Piatetsky-Shapiro, and D. McNeill. Selecting and reporting what is interesting. In U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 495–514. AAAI Press / The MIT Press, 1996.
11. R. S. Michalski, I. Mozetic, J. Hong, and N. Lavrac. The multi-purpose incremental learning system aq15 and its testing application to three medical domains. In *Proceedings of the Fifth National Conference on Artificial Intelligence*, pages 1041–1045, 1986.
12. G. Piatetsky-Shapiro. Discovery, analysis, and presentation of strong rules. In G. Piatetsky-Shapiro and W. J. Frawley, editors, *Knowledge Discovery in Databases*, pages 229–248. AAAI / The MIT Press, 1991.
13. G. Piatetsky-Shapiro, C. Matheus, P. Smyth, and R. Uthurusamy. KDD93: progress and challenges. In *AI magazine*, pages 77–87, Fall 1994.
14. G. Piatetsky-Shapiro and C. J. Matheus. The interestingness of deviations. In *AAAI-84 Workshop on Knowledge Discovery in Databases*, pages 25–36, 1994.
15. J. R. Quinlan. *C4.5: Program for machine learning*. Morgan Kaufmann, 1993.
16. A. Silberschatz and A. Tuzhilin. On subjective measures of interestingness in knowledge discovery. In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, pages 275–281, 1995.
17. R. Srikant and R. Agrawal. Mining quantitative association rules in large relational tables. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, 1996.
18. R. Zembowicz and J. M. Zytkow. From contingency tables to various forms of knowledge in databases. In U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 329–349. AAAI Press / The MIT Press, 1996.

# Minimum Message Length Segmentation

Jonathan J. Oliver<sup>1</sup>, Rohan A. Baxter<sup>2</sup> and Chris S. Wallace<sup>1</sup>  
jono@ultimode.com, rohan@ultimode.com, csw@cs.monash.edu.au

<sup>1</sup> Dept. Computer Science, Monash University, Clayton Vic., Australia

<sup>2</sup> Ultimode Systems, 2560 Bancroft Way #213, Berkeley, CA 94704, USA

**Abstract.** The segmentation problem arises in many applications in data mining, A.I. and statistics, including segmenting time series, decision tree algorithms and image processing. In this paper, we consider a range of criteria which may be applied to determine if some data should be segmented into two or regions. We develop a information theoretic criterion (MML) for the segmentation of univariate data with Gaussian errors. We perform simulations comparing segmentation methods (MML, AIC, MDL and BIC) and conclude that the MML criterion is the preferred criterion. We then apply the segmentation method to financial time series data.

## 1 Introduction

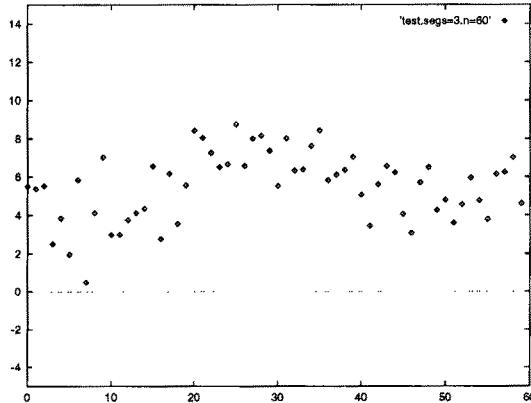
We consider a particular instance of the segmentation problem. The segmentation problem arises wherever it is desired to partition data into distinct homogeneous segments (or regions). The segmentation problem is to decide whether to divide a segment into one or more sub-segments and to choose where to make the divisions.

The segmentation problem arises in applications that partition data in areas such as data mining, A.I. and statistics. The segmentation problem arise in applications such as segmenting time series [14, 16, 5], decision tree algorithms [11, 10], and image processing [7, 6].

### 1.1 The Problem Considered

Here, we consider a univariate problem, where the segment boundaries are defined by *cut-points*. We assume that the data in each segment is defined by a Gaussian distribution. Figure 1 gives an example of the type of data we might consider. We could ask questions such as “Does this data consist of 1, 2 or 3 segments?”; “If it consists of 3 segments, is the behaviour in the first third the same as the behaviour in the last third?” This paper investigates methods for determining for some data:

- (i) how many cut-points should we fit (if any at all)
- (ii) the location of the cut-points, and
- (iii) estimating the parameters (means and variances) for each segment.



**Fig. 1.** Example Data for Segmentation

## 1.2 Motivating the Problem Considered

At first it would appear that the problem as given is overly simple — it would not describe any real world situations, and it should be easy to solve. We argue that these objections are false. Data such as that in Figure 1 might be the number of eye movements per 5 second intervals for a sleeping person, and a doctor may be interested in how many phases of sleep there were, and when they were [14].

A different practical example where this model seems plausible is the incidence of tooth cavities. Previously dentists entertained the burst-remission theory, and dentists spent considerable effort looking for factors that induced remission (i.e., segments with lower means). However, it appears that the data was consistent with the assumption that it was a random walk (i.e. that there was only one segment).

Tong [16] has written a comprehensive book about non linear time series (including segmentation models). We consider such problems in Section 6.

## 1.3 Related Work

The fit of a segmentation model to data can be expressed precisely using maximum likelihood estimation. However, choosing a segmentation model to maximize the likelihood results in a model with homogeneous regions containing only one datum each. Therefore, heuristics for solving the segmentation problem usually involve ‘penalizing’ a segmentation for its model complexity. A number of methods which penalize model complexity are available including AIC [1, 7], BIC [13, 6], Minimum Description Length (MDL) [12] and Minimum Message Length (MML) [17, 18].

In this paper, we extend the MML approach to segmentation offered by Baxter and Oliver [2], to the multiple cutpoint case, and apply the approach to time series problems.

This paper is organised as follows: Section 2 defines the segmentation problem we address here. Section 3 describes a previous MDL approach [4, 10, 11], and describes a shortcoming of this approach. Section 4 gives an MML approach to segmentation. The MML method proposed here differs from the MDL approach by optimising the code for the region boundary *and* including coding penalties for stating the parameters of each region. We then compare a variety of segmentation methods on simulations in Section 5. Section 6 applies the method developed to financial time series problem.

## 2 Notation

Consider some data given as follows. We have  $n$  data points, each of which consists of a pair  $(x_i, y_i)$ . The  $x_i$  are evenly spaced between  $[0, R]$ . The range  $[0, R]$  can be cut into  $C + 1$  pieces by  $C$  segment boundaries (or cutpoints),  $\{v_1, \dots, v_C\}$ . Each  $y_i$  in segment  $j$  is distributed with a Gaussian distribution with mean  $c_j$ , and standard deviation  $\sigma_j$ .

We wish to estimate the following parameters: (i)  $C$ , the number of cutpoints, (ii) the segment boundaries,  $\{v_1, \dots, v_C\}$ , (iii) the means,  $\{c_0, \dots, c_C\}$ , and (iv) the standard deviations,  $\{\sigma_0, \dots, \sigma_C\}$ .

## 3 The Straightforward MDL Approach

Rissanen [12] proposed the straight forward Minimum Description Length (MDL) criterion, which given data  $y$  and parameters  $\theta$  approximates the length as:

$$\text{DescriptionLength}(y, \theta) = -\log f(y|\theta) + \frac{\text{number params}}{2} \log n$$

where  $f(y|\theta)$  is the Gaussian likelihood function,  $-\log f(y|\theta)$  approximates the length of describing the data, and  $\frac{\text{number params}}{2} \log n$  approximates the length of describing the parameter vector. This approximation is unsuited to cutpoint-like parameters. A number of authors [4, 10, 11] have given terms<sup>3</sup> to describe the cost of stating a cutpoint in a message. A straightforward method of coding a cutpoint is to assume that the cutpoint is equally likely to occur in between  $x_i$  and  $x_{i+1}$  for  $i = 1 \dots (n - 1)$  which leads to a cost<sup>4</sup> of  $\log(n)$  to describe the cutpoint. If we wish to state  $C$  cutpoints, then this will require a codeword of length:

$$\text{DescriptionLength}(C \text{ cutpoints}) = \log \binom{n}{C}$$

Dom [4] requires that  $C < \frac{n}{2}$ , otherwise the complexity of the term decreases for increasing  $C$ , which is counter to prior beliefs about segmentation models in most applications.

<sup>3</sup> We note that these authors used this penalty measure in different, but related contexts and that our use of it here is not meant to imply that these authors would advocate its use here.

<sup>4</sup> Most authors simplify matters by allowing the cutpoint to take  $n$  possible values rather than  $n - 1$  values.

### 3.1 A Problem with the Straightforward Approach

A problem with the straightforward MDL approach is that we may use too many bits to describe a cutpoint exactly. Consider a situation where we have the following 17 data points, with points 1-9 been generated by the Gaussian distribution  $N(\mu = 0.0, \sigma^2 = 1.0)$  and points 10-17 been generated by  $N(\mu = 1.0, \sigma^2 = 1.0)$ :

| 1    | 2     | 3     | 4     | 5     | 6    | 7    | 8     | 9    |
|------|-------|-------|-------|-------|------|------|-------|------|
| 2.01 | -1.78 | -1.16 | -2.00 | -1.68 | 0.28 | 0.17 | -0.50 | 0.06 |
| 10   | 11    | 12    | 13    | 14    | 15   | 16   | 17    |      |
| 1.29 | -0.43 | 1.70  | 0.74  | 2.69  | 3.75 | 0.81 | 0.66  |      |

The straightforward MDL approach requires 4 binary bits to describe a cutpoint, The negative log-likelihood  $-\log f(y|\theta)$  is minimised if we place the cutpoint between points 11 and 12. Placing the cutpoint here, results in the following estimates:

$$c_0 = -0.34, \quad c_1 = 1.72, \quad \sigma_0 = 1.22, \quad \sigma_1 = 1.15$$

and a negative log-likelihood:  $-\log f(y|\theta) = 25.68 + 13.50 = 39.18$  bits. The total description length is then:

$$\text{DescriptionLength}(y, \theta) = 39.18 + 8.17 + 4.00 = 51.35 \text{ bits}$$

We should also consider encoding the cutpoints less precisely. For example, we could use an encoding scheme which restricts the cutpoints to every second interval, thus requiring only 3 bits to specify a cutpoint. Using this scheme, and placing the cutpoint between points 8 and 9 results in a description length of  $40.28 + 8.17 + 3.00 = 51.45$  bits.

We can further restrict the possible cutpoints to every fourth interval, thus requiring only 2 bits to specify the cutpoint. Using this scheme, and placing the cutpoint between points 8 and 9 results in a description length of  $40.28 + 8.17 + 2.00 = 50.45$  bits.

Obviously there are many such schemes — the issue we raise is that we may consider schemes where less than 4 bits are required to encode a cutpoint. However, using fewer bits to describe the cutpoint means that our model is less likely to fit the data well.

The MML approach requires us to determine how precisely we wish to state parameters, and hence the mathematics in this paper optimises the choice of coding schemes for cutpoints.

## 4 Applying MML to Segmentation

We consider sending a message for this data of the form:

$$C, \quad c_0, \dots, c_C, \quad \sigma_0, \dots, \sigma_C, \quad v_1, \dots, v_C, \quad y_1, \dots, y_n.$$

The distance between successive  $x_i$  is assumed known. Since the  $x_i$  are evenly spaced, one can work out the number of  $x_i$  in any region from knowing the size of the region. The range of  $x_i$  is assumed to be known by the receiver *a priori*.

#### 4.1 Minimum Message Length Formulas

Wallace and Freeman [18] showed that under some fairly general conditions (a locally flat prior and quadratic log-likelihood function) the expected message length (taking the expectation over coding schemes [8, Section 3.3.1]) for sending  $y$  and parameters  $\theta$  is:

$$E(\text{MessLen}(y, \theta)) = -\log h(\theta) - \log f(y|\theta) + 0.5 \log \det(F(\theta)) + \frac{d}{2} \log \kappa_d + \frac{d}{2}$$

where  $h(\theta)$  is the assumed known prior density on  $\theta$ ,  $d$  is the dimension of  $\theta$ ,  $f(y|\theta)$  is the likelihood, of  $y$  given  $\theta$ ,  $\det(F(\theta))$  is the determinant of the Fisher Information matrix, and  $\kappa_d$  is the  $d$  dimensional lattice constant.

The Wallace and Freeman approximation does not apply to cutpoint-like parameters because the log-likelihood function is not continuous, and hence the Fisher Information matrix is not defined for this type of parameter.

#### 4.2 The One Segment, $C = 0$ , case

For fitting a constant with no cut points  $C = 0$ , our  $\theta$  consists of two parameters,  $c_0$  and  $\sigma_0$ . We choose a non-informative (improper) prior based on the population variance of  $y_i$  [17, 9]:

$$h(c_0, \sigma_0) = \frac{1}{2\sigma_{pop}^2}$$

where  $\sigma_{pop}$  is the standard deviation of the  $y_i$ .

Since the likelihood is Gaussian  $N(c_0, \sigma_0^2)$ , the Fisher Information matrix in this case has two diagonal entries and is:

$$\det(F(c_0, \sigma_0)) = \frac{2n^2}{\sigma_0^4}$$

For a Gaussian likelihood, the negative log-likelihood,  $L_0$  simplifies:

$$L_0 = -\log f(y|\theta) = n \log(\sqrt{2\pi}\sigma_0) + \sum_{i=1}^n \frac{(y_i - c_0)^2}{2\sigma_0^2} = n \log(\sqrt{2\pi}\sigma_0) + \frac{n}{2} \quad (1)$$

Hence, we get the following expression for the expected message length:

$$E(\text{MessLen}) = -\log h(c_0, \sigma_0) + 0.5 \log \det(F(c_0, \sigma_0)) + n \log(\sqrt{2\pi}\sigma_0) + \frac{n}{2} + \frac{\log \kappa_2}{2} + \frac{d}{2}$$

where  $d = 2$  and  $\kappa_2 = \frac{5}{36\sqrt{3}}$  [3].

### 4.3 The $C = 1$ case

We now consider the effect of stating the cut point,  $v$ , imprecisely. Let the cut point have precision  $AOPV_v$  (an acronym for Accuracy Of Parameter Value).

Let  $\epsilon$  be the difference in the  $v$  stated in the message, and the maximum likelihood  $v$  estimated from the data. Assume  $\epsilon$  is uniformly distributed in the range  $[-\frac{AOPV_v}{2}, \frac{AOPV_v}{2}]$ . We now need to state  $c_0$  and  $c_1$ , the constants fitted to the data in the regions on each side of the cut point and also the cut point itself.

In the following we denote the set of  $x_i$  in region 0 fitted by constant  $c_0$  as  $S_0$ . We do the same for the set of  $x_i$  in region 1 fitted by constant  $c_1$ , denoting it  $S_1$ . Let  $n_0$  be the number of items in  $S_0$  and  $n_1$  be the number of items in  $S_1$ . The residual errors are assumed to be distributed as  $N(0, \sigma_0^2)$  for region  $S_0$  and as  $N(0, \sigma_1^2)$  for region  $S_1$ . We assume that the  $v$  is uniformly distributed, and hence  $h(v) = \frac{1}{R}$ . The message length expression for the parameters is then written as follows:

$$\begin{aligned} MessLen(\theta) = & -\log h(c_0, \sigma_0) - \log h(c_1, \sigma_1) - \log 1/R \\ & + 0.5 \log \det(F(c_0, \sigma_0)) + 0.5 \log \det(F(c_1, \sigma_1)) - \log AOPV_v \\ & + 2 + 2 \log \kappa_4 \end{aligned} \quad (2)$$

We note that, given our assumptions about evenly spaced  $x$ , we expect  $n(1 - \frac{|\epsilon|}{R})$  data items will lie in their correct regions, but we expect  $\frac{n|\epsilon|}{R}$  data items will be put in the ‘wrong’ region.

Let  $MLC_j$  be the per item data cost of stating an item *correctly* put in segment  $j$ . Hence,

$$MLC_0 = \log(\sqrt{2\pi}\sigma_0) + \sum_{i \in S_0} \frac{(y_i - c_0)^2}{2\sigma_0^2 n_0}$$

Let  $MLW_j$  be the per item data cost of stating an item *wrongly* put in segment  $j$  and hence,

$$MLW_0 = \log(\sqrt{2\pi}\sigma_1) + \sum_{i \in S_0} \frac{(y_i - c_1)^2}{2\sigma_1^2 n_0}$$

The message length expression for the data is then:

$$MessLen(y|\theta) = MessLen(y \in \text{correct region}|\theta) + MessLen(y \in \text{wrong region}|\theta)$$

which we approximate as:

$$\begin{aligned} MessLen(y|\theta) \approx & n_0 MLC_0 + n_1 MLC_1 - \frac{MLC_0 + MLC_1}{2} \left( \frac{n|\epsilon|}{R} \right) + \\ & \frac{MLW_0 + MLW_1}{2} \left( \frac{n|\epsilon|}{R} \right) \end{aligned}$$

We wish to determine the expected message length. The expected cost of stating incorrectly identified data is simplified by letting  $D = c_0 - c_1$ :

$$E(MLW_0) = \log(\sqrt{2\pi}\sigma_1) + \frac{RSS_0 + n_0 D^2}{2\sigma_1^2 n_0}$$

where  $RSS_0$  is the residual sum of squares ( $RSS_0 = \sum_{i \in S_0} (y_i - c_0)^2$ ).

The expected value of the absolute value of  $\epsilon$  is  $\frac{AOPV_v}{4}$ , since

$$E(|\epsilon|) = \frac{2}{AOPV_v} \int_0^{\frac{AOPV_v}{2}} x dx = \frac{AOPV_v}{4}.$$

Hence, the expected message length for the data is:

$$E(MessLen(y|\theta)) = L_0 + L_1 + \left( \frac{nAOPV_v}{8R} \right) E(MLW_0 - MLC_0 + MLW_1 - MLC_1) \quad (3)$$

where  $L_0$  and  $L_1$  are the negative log likelihoods of segment 0 and segment 1 respectively (as defined in Equation (1)).

We now sum the terms which contain  $AOPV_v$  from Equations (2) and (3):

$$-\log AOPV_v + \left( \frac{nAOPV_v}{8R} \right) E(MLW_0 - MLC_0 + MLW_1 - MLC_1) \quad (4)$$

We take the partial derivative of Expression (4) w.r.t.  $AOPV_v$ , set the result to 0 and solve for the optimal  $AOPV_v$  to minimize the expected message length expression:

$$AOPV_v = \frac{8R/n}{E(MLW_0 - MLC_0 + MLW_1 - MLC_1)}$$

The  $AOPV_v$  can be interpreted as a volume in the parameter space. As  $n_0$  and  $n_1$  grow, we see that the volume decreases because the estimate of  $v$  can be stated more accurately.

#### 4.4 Message Length Expression

To simplify the algebra, let

$$X = E(MLW_0 - MLC_0) + E(MLW_1 - MLC_1),$$

so that the optimal  $AOPV_v$  is  $\frac{8R/n}{X}$ . We substitute the optimal  $AOPV_v$  into the message length expression obtained by summing Equations (2) and (3) and simplifying:

$$\begin{aligned} E(MessLen(y,\theta)) &= -\log h(c_0, \sigma_0) - \log h(c_1, \sigma_1) - \log 1/R \\ &\quad + 0.5 \log \det(F(c_0, \sigma_0)) + 0.5 \log \det(F(c_1, \sigma_1)) - \log AOPV_v \\ &\quad + 2 + 2 \log \kappa_4 + L_0 + L_1 + \frac{X}{X} \end{aligned} \quad (5)$$

## 4.5 Multiple Cutpoints

We now generalise Equation (5) to  $C > 1$  cutpoints. Let  $MLC_j$  be the per item data cost of stating an item *correctly* put in segment  $j$ . Let  $MLW_{j,k}$  be the per item data cost of stating an item from segment  $j$  *wrongly* put into segment  $k$ . For each cutpoint ( $j = 1..C$ ) let

$$X_j = E(MLW_{j-1,j} - MLC_{j-1}) + E(MLW_{j,j-1} - MLC_j)$$

so that the optimal  $AOPV_{vj}$  for cutpoint  $j$  is:

$$AOPV_{vj} = \frac{8R/(n_{j-1} + n_j)}{X_j}$$

With  $C > 1$  cutpoints, we have:

$$\begin{aligned} E(MessLen(y, \theta)) = & - \sum_{j=0}^C \log h(c_j, \sigma_j) - C \log 1/R + 0.5 \sum_{j=0}^C \log \det(F(c_j, \sigma_j)) \\ & - \log C! - \sum_{j=1}^C \log AOPV_{vj} + \frac{d}{2} + \frac{d}{2} \log \kappa_d + \sum_{j=0}^C L_j + C \end{aligned} \quad (6)$$

## 5 Simulation Results

We ran simulations comparing the following criteria:

- (i) MML, using Equation (6) of this paper.
- (ii) AIC, using  $-\log f(y|\theta) + \text{number params}$  [7].
- (iii) BIC, using  $-\log f(y|\theta) + \frac{\text{number params}}{2} \log n$  [6].
- (iv) MDL, using  $-\log f(y|\theta) + \frac{\text{continuous params}}{2} \log n + \log \binom{n}{C}$ .

### 5.1 The Search Method

It is impractical to consider every possible segmentation of data once we consider multiple cutpoints. We therefore used the following search method. Given a set of data, we consider every binary segmentation (i.e., one cutpoint) and identify those cutpoints which are local maxima in likelihood. We then perform an exhaustive search of segmentations using the cutpoints which are local maxima in likelihood. The segmentations are also required to have a minimum segment length of 3.

| $\hat{k}$ | 1   | 2  | 3  | 4  | 5  | Av. KL |
|-----------|-----|----|----|----|----|--------|
| n=20      |     |    |    |    |    |        |
| MML       | 99  | 0  | 1  | 0  | 0  | 0.085  |
| AIC       | 39  | 35 | 22 | 4  | 0  | 23.926 |
| BIC       | 78  | 15 | 7  | 0  | 0  | 23.058 |
| MDL       | 92  | 5  | 3  | 0  | 0  | 20.238 |
| n=40      |     |    |    |    |    |        |
| MML       | 98  | 2  | 0  | 0  | 0  | 0.033  |
| AIC       | 30  | 20 | 31 | 14 | 5  | 9.089  |
| BIC       | 87  | 10 | 3  | 0  | 0  | 7.487  |
| MDL       | 98  | 2  | 0  | 0  | 0  | 0.424  |
| n=80      |     |    |    |    |    |        |
| MML       | 99  | 0  | 0  | 0  | 1  | 0.020  |
| AIC       | 12  | 9  | 30 | 25 | 24 | 4.446  |
| BIC       | 95  | 4  | 1  | 0  | 0  | 0.483  |
| MDL       | 99  | 1  | 0  | 0  | 0  | 0.265  |
| n=160     |     |    |    |    |    |        |
| MML       | 99  | 1  | 0  | 0  | 0  | 0.007  |
| AIC       | 6   | 9  | 23 | 31 | 31 | 3.961  |
| BIC       | 99  | 1  | 0  | 0  | 0  | 0.088  |
| MDL       | 100 | 0  | 0  | 0  | 0  | 0.007  |

**Table 1.** (a) True no. of segments = 1

| $\hat{k}$ | 1  | 2  | 3  | 4  | 5  | Av. KL |
|-----------|----|----|----|----|----|--------|
| n=20      |    |    |    |    |    |        |
| MML       | 69 | 28 | 3  | 0  | 0  | 0.324  |
| AIC       | 15 | 47 | 30 | 8  | 0  | 24.172 |
| BIC       | 48 | 38 | 9  | 5  | 0  | 23.510 |
| MDL       | 70 | 21 | 6  | 3  | 0  | 23.061 |
| n=40      |    |    |    |    |    |        |
| MML       | 37 | 60 | 3  | 0  | 0  | 0.140  |
| AIC       | 4  | 40 | 32 | 21 | 3  | 13.559 |
| BIC       | 29 | 58 | 12 | 1  | 0  | 12.412 |
| MDL       | 53 | 41 | 6  | 0  | 0  | 10.166 |
| n=80      |    |    |    |    |    |        |
| MML       | 11 | 81 | 6  | 1  | 1  | 0.088  |
| AIC       | 0  | 17 | 27 | 30 | 26 | 7.246  |
| BIC       | 16 | 76 | 7  | 0  | 1  | 0.816  |
| MDL       | 34 | 63 | 3  | 0  | 0  | 0.770  |
| n=160     |    |    |    |    |    |        |
| MML       | 0  | 98 | 2  | 0  | 0  | 0.025  |
| AIC       | 0  | 23 | 32 | 26 | 19 | 2.777  |
| BIC       | 1  | 97 | 2  | 0  | 0  | 0.108  |
| MDL       | 2  | 98 | 0  | 0  | 0  | 0.027  |

**Table 1.** (b) True no. of segments = 2

| $\hat{k}$ | 1  | 2  | 3  | 4  | 5 | 6 | Av. KL |
|-----------|----|----|----|----|---|---|--------|
| n=20      |    |    |    |    |   |   |        |
| MML       | 31 | 65 | 4  | 0  | 0 | 0 | 0.320  |
| AIC       | 3  | 49 | 43 | 4  | 1 | 0 | 17.441 |
| BIC       | 15 | 61 | 22 | 2  | 0 | 0 | 16.884 |
| MDL       | 34 | 52 | 13 | 1  | 0 | 0 | 16.034 |
| n=40      |    |    |    |    |   |   |        |
| MML       | 3  | 85 | 12 | 0  | 0 | 0 | 0.191  |
| AIC       | 0  | 28 | 41 | 26 | 4 | 1 | 10.379 |
| BIC       | 3  | 79 | 16 | 1  | 1 | 0 | 9.337  |
| MDL       | 10 | 78 | 10 | 1  | 1 | 0 | 9.255  |

| $\hat{k}$ | 1 | 2  | 3  | 4  | 5  | 6  | Av. KL |
|-----------|---|----|----|----|----|----|--------|
| n=80      |   |    |    |    |    |    |        |
| MML       | 0 | 50 | 50 | 0  | 0  | 0  | 0.106  |
| AIC       | 0 | 4  | 34 | 35 | 23 | 4  | 6.089  |
| BIC       | 0 | 61 | 36 | 3  | 0  | 0  | 2.786  |
| MDL       | 0 | 77 | 22 | 1  | 0  | 0  | 1.358  |
| n=160     |   |    |    |    |    |    |        |
| MML       | 0 | 8  | 92 | 0  | 0  | 0  | 0.044  |
| AIC       | 0 | 0  | 32 | 28 | 21 | 19 | 2.729  |
| BIC       | 0 | 21 | 79 | 0  | 0  | 0  | 1.416  |
| MDL       | 0 | 46 | 54 | 0  | 0  | 0  | 1.316  |

**Table 2.** True no. of segments = 3

## 5.2 Results

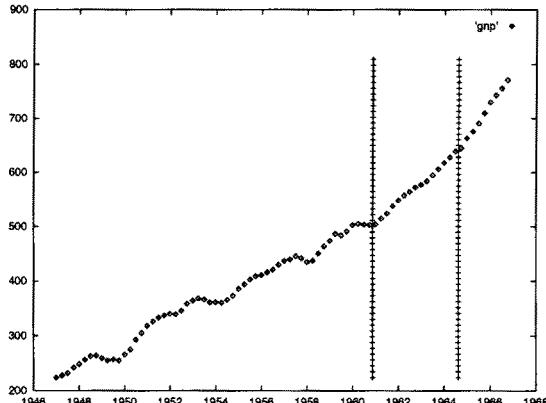
In Tables 1(a), 1(b) and 2, we give the results when we presented simulated data to the criteria given in Section 5. The data used in the simulations was generated according to the following distributions:

- Table 1(a) — One segment with distribution  $N(\mu = 0, \sigma^2 = 1)$ ,
- Table 1(b) — Two segments with the first half distributed as  $N(\mu = 0, \sigma^2 = 1)$  and the second half distributed as  $N(\mu = 1, \sigma^2 = 1)$ , and
- Table 2 — Three segments with the first third distributed as  $N(\mu = 0, \sigma^2 = 1)$ ,

the middle third distributed as  $N(\mu = 1, \sigma^2 = 1)$  and the last third distributed as  $N(\mu = 2, \sigma^2 = 1)$ .

In each simulation, we generated  $n$  points from the appropriate distribution. We applied the search method described in Section 5.1. We applied the criteria from Section 5 and listed the number of times the criteria estimated each value of  $k$  from 100 simulations. Tables 1(a), 1(b) and 2 also give the average Kullback-Liebler distance (Av. KL) between the predicted distribution, and the underlying distribution<sup>5</sup>.

## 6 Time Series Applications



**Fig. 2.** The US GNP 1947 – 1966

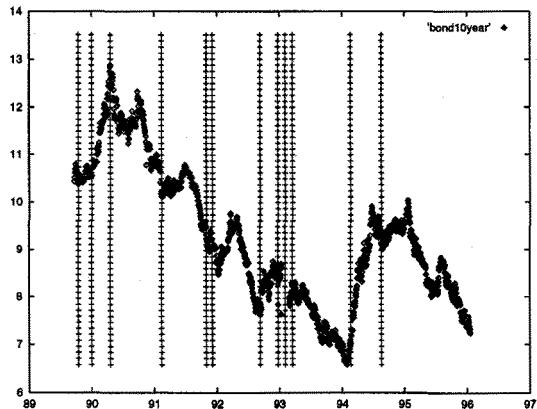
We may model time series of the form:  $z_{t+1} = z_t + c_j + \epsilon(0, \sigma_j^2)$  by setting  $y_t = z_{t+1} - z_t$ . This may be a reasonable method for segmenting data from examples such as: (i) economic time series, (ii) electrocardiogram measurements and (iii) eye movement measurements from a sleeping person.

We segmented the quarterly gross national product (GNP) for the United States from 1947 – 1966 [14]. Figure 2<sup>6</sup> shows the preferred MML segmentation for this data. The BIC and MDL criteria also preferred this segmentation, while the AIC criterion preferred a segmentation with 7 segments.

<sup>5</sup> The Kullback-Liebler distance (given for example in [15, Chp. 9]) between a true distribution  $N(\mu_t, \sigma_t^2)$  and a fitted distribution  $N(\mu_f, \sigma_f^2)$  is

$$\log \frac{\sigma_f}{\sigma_t} - \frac{1}{2} + \frac{1}{2\sigma_f^2} (\sigma_t^2 + (\mu_t - \mu_f)^2).$$

<sup>6</sup> The units in the figure are billions of (non constant) dollars.



**Fig. 3.** The Canadian 10 year bond yield 1989 – 1996 with 12 cut points

We then considered segmenting a larger data set, namely the Canadian 10 year bond yield. The data set consists of 1514 values of the Canadian 10 year bond (measured in Canadian dollars) for the period 1989 – 1996. The segmentation program took 24 minutes and 31 seconds to examine segmentations of up to 30 segments on a DECstation 5000/20 using a greedy search strategy. The MML criterion found evidence for there being at least 8 cut points since the message length of the data with no cut points was 5501.9 nits and the message length with 8 cut points was 5295.1 nits. The minimum message length (with 12 cut points – see Figure 3) was 5282.8 nits.

## 7 Conclusion

We have derived a message length criterion for the segmentation of univariate data with Gaussian noise. We tested the criterion and found that it outperformed other criteria (AIC, BIC, MDL) in determining the number of regions in the simulations conducted here. Of the methods considered in this paper, the average Kullback-Liebler distance between the fitted distribution and true distribution was far smaller for the MML method. The method was successfully applied to two financial time series problems; the method scaled up reasonably to handle a data set with 1514 data points.

## Acknowledgments

We would like to thank Catherine Forbes, David Albrecht and Wray Buntine for valuable discussions, and the anonymous referees for valuable critical comments. Jon Oliver acknowledges research support by Australian Research Council (ARC) Postdoctoral Research Fellowship F39340111.

## References

1. H. Akaike. Information theory and an extension of the maximum likelihood principle. In B.N. Petrov and F. Csaki, editors, *Proc. of the 2nd International Symposium on Information Theory*, pages 267–281, 1973.
2. R.A. Baxter and J.J. Oliver. The kindest cut: minimum message length segmentation. In S. Arikawa and A. Sharma, editors, *Lecture Notes in Artificial Intelligence 1160, Algorithmic Learning Theory, ALT-96*, pages 83–90, 1996.
3. J.H. Conway and N.J.A Sloane. *Sphere Packings, Lattices and Groups*. Springer-Verlag, London, 1988.
4. B. Dom. MDL estimation with Small Sample Sizes including an application to the problem of segmenting binary strings using bernoulli models. Technical Report RJ 9997 (89085) 12/15/95, IBM Research Division, Almaden Research Center, 650 Harry Rd, San Jose, CA, 95120-6099, 1995.
5. G. Koop and S.M. Potter. Bayes Factors and nonlinearity: Evidence from economic time series. UCLA Working Paper, August 1995, submitted to *Journal of Econometrics*.
6. Mengxiang Li. Minimum description length based 2-D shape description. In *IEEE 4th Int. Conf. on Computer Vision*, pages 512–517, May 1992.
7. Z. Liang, R.J. Jaszcak, and R.E. Coleman. Parameter estimation of finite mixtures using the EM algorithm and information criteria with applications to medical image processing. *IEEE Trans. on Nuclear Science*, 39(4):1126–1133, 1992.
8. J.J. Oliver and D.J. Hand. Introduction to minimum encoding inference. Technical report TR 4-94, Dept. of Statistics, Open University, Walton Hall, Milton Keynes, MK7 6AA, UK, 1994. Available on the WWW from <http://www.cs.monash.edu.au/~jono>.
9. J.J. Oliver, Baxter R.A., and Wallace C.S. Unsupervised Learning using MML. In *Machine Learning: Proc. of the Thirteenth International Conference (ICML 96)*, pages 364–372. Morgan Kaufmann Publishers, San Francisco, CA, 1996. Available on the WWW from <http://www.cs.monash.edu.au/~jono>.
10. B. Pfahringer. Compression-based discretization of continuous attributes. In *Machine Learning: Proc. of the Twelfth International Workshop*, pages 456–463, 1995.
11. J.R. Quinlan. Improved use of continuous attributes in C4.5. *Journal of Artificial Intelligence*, 4:77–90, 1996.
12. J. Rissanen. Modeling by shortest data description. *Automatica*, 14:465–471, 1978.
13. G. Schwarz. Estimating dimension of a model. *Ann. Stat.*, 6:461–464, 1978.
14. S.L. Sclove. On segmentation of time series. In S. Karlin, T. Amemiya, and L. Goodman, editors, *Studies in econometrics, time series, and multivariate statistics*, pages 311–330. Academic Press, 1983.
15. C.W. Therrien. *Decision, estimation, and classification : an introduction to pattern recognition and related topics*. Wiley, New York, 1989.
16. H. Tong. *Non-linear time series : a dynamical system approach*. Clarendon Press, Oxford, 1990.
17. C.S. Wallace and D.M. Boulton. An information measure for classification. *Computer Journal*, 11:185–194, 1968.
18. C.S. Wallace and P.R. Freeman. Estimation and inference by compact coding. *Journal of the Royal Statistical Society (Series B)*, 49:240–252, 1987.

# Bayesian Classification Trees with Overlapping Leaves Applied to Credit-Scoring

Gerhard Paass and Jörg Kindermann

RWCP Theoretical Foundation GMD Laboratory,  
D-53754 Sankt Augustin, Germany

**Abstract.** We develop a Bayesian procedure for classification with trees by switching between different model structures. For classification trees with overlap we use a Markov chain Monte Carlo procedure to produce an ensemble of trees which allow the assessment of prediction uncertainty and the value of new information. The approach is applied to a large credit scoring application.

## 1 Introduction

A large number of classification procedures have been developed in statistics, data-mining and neural networks [5]. Usually the feature vector  $\mathbf{x}$  defines the conditional probability  $p(y=1|\mathbf{x}) \in [0, 1]$  of the class variable  $y$ . Especially appealing are *local* approaches, which cover the input space  $\mathcal{X}$  with a finite set  $\mathcal{X}_\tau$  of local regions such that  $\mathcal{X} = \bigcup_\tau \mathcal{X}_\tau$ . Within each region the function  $f(\mathbf{x}) := p(y=1|\mathbf{x})$  is approximated by a simple local function  $\hat{f}_\tau(\mathbf{x})$ , e.g. a constant, a linear or quadratic polynomial. This is motivated by Taylor's theorem which states that if a local region is small enough any continuous function can be well approximated by a low order polynomial within it.

*Classification trees* [2, 3] are local models which recursively partitions the input space into a number of disjunct regions  $\mathcal{X}_\tau$  and separately train a model  $f_\tau(\mathbf{x})$  in each region. The regions are formed in such a way that the approximation error is minimized. However, as each region requires a number of observations to estimate the local function relatively few regions can be formed and the approximation error gets large near the boundaries.

In this paper we propose a tree based classification procedure which borrows “estimation power” from nearby regions:

- We allow some *overlap* between the regions. Observations located in this overlap are assigned to both regions for training as well as for prediction. This increases the average number of observations in each final region.
- Not a single ‘optimal’ tree is determined in a greedy fashion, but using *Bayesian* statistics a large number of plausible trees is constructed. Each tree has a different set of local regions with different boundaries. Because the prediction is determined as the average prediction of the trees the biases near boundaries are potentially reduced.

Bayesian statistics derives a probability measure on the model parameters, which describes the ‘plausibility’ that a model with this parameter has generated the data [1]. Tree models involve an extra complication as the number of parameters varies as the tree grows. Hence we require a procedure which is able to switch between different model structures. We develop a Bayesian method for classification trees with overlap and implement the corresponding Markov chain Monte Carlo procedure as a variant of [4]. Overlapping trees generated in a greedy fashion have been proposed by [7].

In the next section we introduce the basic concepts of Bayesian classification and the Metropolis procedure which is used to obtain a representative set of models. The third section introduces an equilibrium theorem for switching between models of different complexity. Section 4 describes the notion of Bayesian classification trees with overlapping leaves and the generation of a Markov chain of trees. In section 5 we describe our credit scoring application and derive the value of new information similar to [11]. Section 6 compares our approach with alternative models for credit scoring data.

## 2 Bayesian Classification

### 2.1 Basic Concepts

Assume for each object we have features  $\mathbf{x} = (x_1, \dots, x_k) \in \mathcal{X}$  and want to determine the class  $y \in \{0, 1\}$  of the object. The relation between  $\mathbf{x}$  and  $y$  is described by a conditional distribution  $p(y|\mathbf{x}, \mathbf{w})$  with unknown parameters  $\mathbf{w} \in \mathcal{W}$ . Suppose we have a sample  $(\mathbf{x}^{(1)}, y^{(1)}), \dots, (\mathbf{x}^{(n)}, y^{(n)})$ , where  $y^{(i)}$  is independently distributed according to the ‘true’  $p(y|\mathbf{x}^{(i)})$ . We denote the input data by  $\mathbf{X} := (\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(n)})$  and the output data by  $\mathbf{y} := (y^{(1)}, \dots, y^{(n)})$ . Let  $p(\mathbf{w})$  be our *prior* distribution of the model parameters describing the relative plausibility of parameter values *before* any data is available. For fixed data  $\mathbf{X}$  and  $\mathbf{y}$  the *likelihood* of  $\mathbf{w}$  is defined as  $p(\mathbf{y}|\mathbf{X}, \mathbf{w}) := \prod_{i=1}^n p(y^{(i)}|\mathbf{x}^{(i)}, \mathbf{w})$ . Then the *Bayesian formula* yields the *posterior distribution*

$$p(\mathbf{w}|\mathbf{y}, \mathbf{X}) = \frac{p(\mathbf{y}|\mathbf{X}, \mathbf{w}) p(\mathbf{w})}{\int p(\mathbf{y}|\mathbf{X}, \mathbf{w}) p(\mathbf{w}) d\mathbf{w}} \quad (1)$$

It describes the relative plausibility of different parameters  $\mathbf{w}$  after the data  $\mathbf{X}$  and  $\mathbf{y}$  has been observed [1].

Let  $q_c$  be the probability that  $y$  has the class-value  $c$ . For a fixed input  $\mathbf{x}$  the ‘parameter’  $q_c$  is uncertain and has a probability distribution determined by the distribution of  $\mathbf{w}$ . We can compute the probability  $P(q_c \leq \eta | \mathbf{x}, \mathbf{y}, \mathbf{X})$  as  $\int_{\{\mathbf{w} | p(y=c|\mathbf{x}, \mathbf{w}) \leq \eta\}} p(\mathbf{w}|\mathbf{y}, \mathbf{X}) d\mathbf{w}$ . The ‘average’ probability that a new object with features  $\mathbf{x}$  belongs to class  $c$  is

$$E(q_c | \mathbf{x}, \mathbf{y}, \mathbf{X}) = \int p(y=c|\mathbf{x}, \mathbf{w}) p(\mathbf{w}|\mathbf{y}, \mathbf{X}) d\mathbf{w} \quad (2)$$

Often we face the problem of taking an action  $a \in \mathcal{A}$ , whose consequences depend on the class  $c$  of an object. In the case of credit-scoring  $a$  may be ‘grant

credit' or 'deny credit'. Let  $L(a; c) \in \Re$  be the loss we incur, if action  $a$  is taken and  $c$  is the actual class of the object. The *expected loss*  $E(L(a)|\mathbf{x}, \mathbf{y}, \mathbf{X})$  of  $a$  is

$$\int \sum_{y=0}^1 L(a; y) p(y|\mathbf{x}, \mathbf{w}) p(\mathbf{w}|\mathbf{y}, \mathbf{X}) d\mathbf{w} = \sum_{y=0}^1 L(a; y) E(q_y|\mathbf{x}, \mathbf{y}, \mathbf{X}) \quad (3)$$

According to the Bayesian decision theory [1] it is optimal to select the action  $a$  with *minimal expected loss*. Note that only the mean value of  $q_c$  enters the decision, not its variance.

## 2.2 The Metropolis-Hastings Algorithm

As (2) and (3) in general cannot be evaluated analytically, we have to perform a Monte-Carlo analysis. This involves the construction of a Markov chain  $\mathbf{w}(0), \mathbf{w}(1), \dots$  designed to be distributed according to the posterior density  $\pi(\mathbf{w}) = p(\mathbf{w}|\mathbf{y}, \mathbf{X})$ . If the chain is currently at  $\mathbf{w} = \mathbf{w}(t)$ , the *Hastings algorithm* [13] requires a *proposal density*  $q^\rightarrow(\mathbf{w}, \tilde{\mathbf{w}})$ , which is the conditional distribution of proposing a move from  $\mathbf{w}$  to  $\tilde{\mathbf{w}}$ . The *acceptance probability* is defined as [10]

$$p^{acc}(\mathbf{w}, \tilde{\mathbf{w}}) = \min \left\{ \frac{p(\mathbf{y}|\mathbf{X}, \tilde{\mathbf{w}}) p(\tilde{\mathbf{w}}) q^\rightarrow(\tilde{\mathbf{w}}, \mathbf{w})}{p(\mathbf{y}|\mathbf{X}, \mathbf{w}) p(\mathbf{w}) q^\rightarrow(\mathbf{w}, \tilde{\mathbf{w}})}, 1 \right\} \quad (4)$$

if  $p(\mathbf{y}|\mathbf{X}, \mathbf{w}) p(\mathbf{w}) q^\rightarrow(\mathbf{w}, \tilde{\mathbf{w}}) > 0$ , and  $p^{acc}(\mathbf{w}, \tilde{\mathbf{w}}) = 1$  otherwise. With probability  $p^{acc}(\mathbf{w}, \tilde{\mathbf{w}})$  the candidate  $\tilde{\mathbf{w}}$  is accepted and the chain moves to  $\mathbf{w}(t+1) = \tilde{\mathbf{w}}$ . Otherwise the candidate is rejected and  $\mathbf{w}(t+1)$  takes the old value  $\mathbf{w}$ .

The transition probability is defined as  $p^\rightarrow(\mathbf{w}, \tilde{\mathbf{w}}) := q^\rightarrow(\mathbf{w}, \tilde{\mathbf{w}}) p^{acc}(\mathbf{w}, \tilde{\mathbf{w}})$  if  $\mathbf{w} \neq \tilde{\mathbf{w}}$  and  $p^\rightarrow(\mathbf{w}, \tilde{\mathbf{w}}) = 0$  if  $\mathbf{w} = \tilde{\mathbf{w}}$ . Then the *detailed balance* condition holds

$$\pi(\mathbf{w}) p^\rightarrow(\mathbf{w}, \tilde{\mathbf{w}}) = \pi(\tilde{\mathbf{w}}) p^\rightarrow(\tilde{\mathbf{w}}, \mathbf{w}) \quad (5)$$

If the resulting Markov chain is aperiodic and irreducible (i.e. reaches all states with positive probability) the Markov chain approaches an invariant stationary limit distribution, which is just the posterior distribution  $p(\mathbf{w}|\mathbf{y}, \mathbf{X})$  [13].

## 3 Switching Model Spaces

Suppose we have a countable collection of candidate models  $\{M_v, v \in \mathcal{K}\}$ . Model  $M_v$  has a vector  $\mathbf{w} \in \Re^{n_v}$  of unknown parameters. The dimension  $n_v$  may vary from model to model and the components of  $\mathbf{w}$  may have different interpretations for each model  $M_v$ . In this case the Hastings algorithm is not directly applicable.

The following derivation is an adaption of [9]. Assuming that the input data alone contains no information on  $\mathbf{w}$  and  $v$ , we get  $p(\mathbf{w}|v, \mathbf{x}) = p(\mathbf{w}|v)$  and  $p(v|\mathbf{x}) = p(v)$  and therefore  $p(\mathbf{y}, \mathbf{w}, v, \mathbf{x}) = p(\mathbf{y}|\mathbf{w}, v, \mathbf{x}) p(\mathbf{w}|v) p(v) p(\mathbf{x})$ . Generally  $(v, \mathbf{w})$  varies over  $\mathcal{W} := \bigcup_{v \in \mathcal{K}} \mathcal{W}_v$ , where  $\mathcal{W}_v := \{v\} \times \Re^{n_v}$ . For fixed data we denote the posterior distribution by  $\pi(v, \mathbf{w}) := p(v, \mathbf{w}|\mathbf{y}, \mathbf{X})$ .

We assume that the  $m$ -th move takes all elements of a specific subspace  $\mathcal{W}_v$  to exactly one subspace  $\mathcal{W}_{\tilde{v}}$  and vice versa. Let  $\vec{q}_m((v, \mathbf{w}), d(\tilde{v}, \tilde{\mathbf{w}}))$  be the sub-probability distribution of proposing a change of  $(v, \mathbf{w}) \in \mathcal{W}_v$  to  $(\tilde{v}, \tilde{\mathbf{w}}) \in \mathcal{W}_{\tilde{v}}$  if move  $m$  is selected. There are two types of moves. The first type transforms  $(v, \mathbf{w}) \in \mathcal{W}_v$  into some  $(\tilde{v}, \tilde{\mathbf{w}}) \in \mathcal{W}_{\tilde{v}}$ . The second move type switches between two different specific subspaces  $\mathcal{W}_v$  and  $\mathcal{W}_{\tilde{v}}$ . As the dimensions of parameter vectors differ we compensate this by including new vectors  $\mathbf{u}$  and  $\tilde{\mathbf{u}}$  with known distributions. Let  $\mathbf{u} \in \mathcal{U}_v \subseteq \Re^{m_v}$  and  $\tilde{\mathbf{u}} \in \mathcal{U}_{\tilde{v}} \subseteq \Re^{m_{\tilde{v}}}$  be two independent continuous random variables generated according to known proper densities  $p_m(\mathbf{u})$  and  $p_m(\tilde{\mathbf{u}})$ . The dimensions have to satisfy  $n_v + m_v = n_{\tilde{v}} + m_{\tilde{v}}$ . Let  $h_m : (\mathbf{u}, \mathbf{w}) \rightarrow (\tilde{\mathbf{u}}, \tilde{\mathbf{w}})$  be a differentiable bijective map.

To switch from  $(v, \mathbf{w}) \in \mathcal{W}_v$  to state  $(\tilde{v}, \tilde{\mathbf{w}}) \in \mathcal{W}_{\tilde{v}}$ , a vector  $\mathbf{u} \sim p_m(\mathbf{u})$  is independently generated. The new state is defined as  $(\tilde{v}, \tilde{\mathbf{w}})$ , (6) with  $\tilde{\mathbf{w}}$  determined as  $(\tilde{\mathbf{u}}, \tilde{\mathbf{w}}) = h_m(\mathbf{u}, \mathbf{w})$ .

To switch from  $(\tilde{v}, \tilde{\mathbf{w}}) \in \mathcal{W}_{\tilde{v}}$  back to state  $(v, \mathbf{w}) \in \mathcal{W}_v$ , a vector  $\tilde{\mathbf{u}} \sim p_m(\tilde{\mathbf{u}})$  is independently generated. The new state is defined as  $(v, \mathbf{w})$ , (7) with  $\mathbf{w}$  generated by the inverse mapping  $(\mathbf{u}, \mathbf{w}) = h_m^{-1}(\tilde{\mathbf{u}}, \tilde{\mathbf{w}})$ .

Note that the dimensions  $m_v$  or  $m_{\tilde{v}}$  may be zero. Define  $\tilde{\mathbf{w}}(\mathbf{u}, \mathbf{w}) := \tilde{\mathbf{w}}$  if  $(\tilde{\mathbf{u}}, \tilde{\mathbf{w}}) = h_m(\mathbf{u}, \mathbf{w})$ . We get  $\vec{q}_m((v, \mathbf{w}), \mathcal{W}_s) = 0$  for  $s \neq \tilde{v}$ , if the  $m$ -th move switches model spaces. The probability of choosing move  $m$  in state  $(v, \mathbf{w})$  is defined as  $j_m(v, \mathbf{w}) := \vec{q}_m((v, \mathbf{w}), \mathcal{W}_{\tilde{v}})$ . The probability of switching from  $(v, \mathbf{w}) \in \mathcal{W}_v$  to  $(\tilde{v}, \tilde{A}) \subseteq \mathcal{W}_{\tilde{v}}$  is

$$\vec{q}_m((v, \mathbf{w}), (\tilde{v}, \tilde{A})) = j_m(v, \mathbf{w}) * p_m(U) \quad (8)$$

where  $U = \{\mathbf{u} : \tilde{\mathbf{w}}_m(\mathbf{u}, \mathbf{w}) \in \tilde{A}\}$ . With probability  $\sum_m \vec{q}_m((v, \mathbf{w}), \mathcal{W})$  the Markov chain moves away from  $(v, \mathbf{w})$ , and no change to the present state is proposed with probability  $1 - \sum_m \vec{q}_m((v, \mathbf{w}), \mathcal{W})$ . Note that for many states the probability of choosing a specific move  $m$  may be 0.

**Theorem 1. Equilibrium Distribution** Consider move  $m$  switching between states  $(v, \mathbf{w}) \in \mathcal{W}_v$  and  $(\tilde{v}, \tilde{\mathbf{w}}) \in \mathcal{W}_{\tilde{v}}$ . Let  $j_m(v, \mathbf{w})$ ,  $p_m(\mathbf{u})$ , and  $h_m(\mathbf{u}, \mathbf{w})$  be defined as in (6). Let  $p(v)$  be the prior probability of model  $M_v$  and  $p(\mathbf{w}|v)$  be the proper prior density of the parameters  $\mathbf{w}$  of model  $M_v$ . Denote the Jacobian determinant of the mapping  $h_m$  by  $\left| \frac{\partial h_m(\mathbf{u}, \mathbf{w})}{\partial (\mathbf{u}, \mathbf{w})} \right|$ . Define the acceptance probability of move  $m$  by

$$p_m^{acc}((v, \mathbf{w}), (\tilde{v}, \tilde{\mathbf{w}})) = \min \left( 1, \left| \frac{\partial h_m(\mathbf{u}, \mathbf{w})}{\partial (\mathbf{u}, \mathbf{w})} \right| * \frac{p(\mathbf{y}|\mathbf{X}, \tilde{v}, \tilde{\mathbf{w}}) p(\tilde{\mathbf{w}}|\tilde{v}) p(\tilde{v}) j_m(\tilde{v}, \tilde{\mathbf{w}}) p_m(\tilde{\mathbf{u}})}{p(\mathbf{y}|\mathbf{X}, v, \mathbf{w}) p(\mathbf{w}|v) p(v) j_m(v, \mathbf{w}) p_m(\mathbf{u})} \right) \quad (9)$$

Then the equilibrium distribution of the resulting Markov chain is the posterior distribution  $\pi(v, \mathbf{w}) = p(v, \mathbf{w}|\mathbf{y}, \mathbf{X})$ .

The proof [9] shows that the detailed balance condition (5) holds for the resulting equilibrium distribution. The densities have to be known only up to a common *common* factor  $\sum_v p(v) \int p(\mathbf{y}|v, \mathbf{w}, \mathbf{X}) p(\mathbf{w}|v) d\mathbf{w}$ , which cancels out in (9). This is similar to the usual Hastings formula (4).

## 4 Bayesian Classification Trees

### 4.1 Overlapping Regions

Tree models are a flexible method for specifying the conditional distribution of a dependent variable  $y$ , given a vector  $\mathbf{x} = (x_1, \dots, x_k) \in \mathcal{X}$  of predictor values. They divide the predictor space  $\mathcal{X}$  into rectangular regions  $\mathcal{X}_\tau$  by recursive splits of  $\mathcal{X}$  and assume that the conditional distribution of  $y$  within the terminal regions  $\mathcal{X}_\tau$  is identical for all  $\mathbf{x}$  within  $\mathcal{X}_\tau$ .

In this paper we recursively divide an existing region  $\mathcal{X}_\tau$  into two new regions  $\mathcal{X}_{\tau_1}$  and  $\mathcal{X}_{\tau_2}$  which are not disjoint, but may have some overlap. We select a variable  $x_{s_\tau}$  with the *split index*  $s_\tau \in \{1, \dots, k\}$  and define

$$\mathcal{X}_{\tau_1} = \{\mathbf{x} \in \mathcal{X}_\tau | x_{s_\tau} \leq \xi_\tau^+\} \quad \mathcal{X}_{\tau_2} = \{\mathbf{x} \in \mathcal{X}_\tau | x_{s_\tau} > \xi_\tau^-\} \quad (10)$$

where the *upper split point*  $\xi_\tau^+$  is greater or equal to the *lower split point*  $\xi_\tau^-$ . A recursive application of this procedure yields a binary tree structure. The non-split terminal regions form the set of *leaves* and cover the input space  $\mathcal{X}$ .

A Bayesian model has to specify completely, how the data is generated from the underlying model. If the tree consists of a single leaf  $\mathcal{X}_\tau$  the dependent variable  $y$  is assumed to be binomially distributed with parameter  $\boldsymbol{\theta}_\tau = (\theta_{\tau,0}, \theta_{\tau,1})$ , where

$$\theta_{\tau,i} = p(y=i | \mathbf{x} \in \mathcal{X}_\tau) \quad (11)$$

If the tree consists of two overlapping leaves  $\mathcal{X}_{\tau_1}$  and  $\mathcal{X}_{\tau_2}$ , then in  $\mathcal{X}_{\tau_1} \setminus \mathcal{X}_{\tau_2}$  the class variable  $y$  again is assumed to be binomially distributed with parameter  $\boldsymbol{\theta}_{\tau_1}$  and in  $\mathcal{X}_{\tau_2} \setminus \mathcal{X}_{\tau_1}$   $y$  is assumed to be binomially distributed with parameter  $\boldsymbol{\theta}_{\tau_2}$ . In the overlap it follows a binomial ‘mixture’ distribution with parameter  $(\boldsymbol{\theta}_{\tau_1} + \boldsymbol{\theta}_{\tau_2})/2$ . This scheme is recursively applied if a region  $\mathcal{X}_\tau$  is covered by two subregions  $\mathcal{X}_{\tau_1}$  and  $\mathcal{X}_{\tau_2}$ . For prediction this has the interesting consequence, that a new observation  $\mathbf{x}$  may belong to more than one leaf and the predicted class probability is a mixture of the separate predictions of each leaf.

If the two split points  $\xi_\tau^-$  and  $\xi_\tau^+$  are identical for all  $\tau$ , we get the usual binary trees as a special case. As an alternative to the constant class probability  $\boldsymbol{\theta}_\tau$ , more complex distributions within the regions can be considered, e.g. multinomial distributions or generalized linear models. In any case the number of parameters is increased as the tree grows.

### 4.2 Prior and Posterior Distributions

As prior density  $p(\boldsymbol{\theta})$  for the binomial parameter we use the Dirichlet density  $\text{Di}(\boldsymbol{\alpha})$  [12] defined as  $p(\boldsymbol{\theta}|\boldsymbol{\alpha}) = \frac{1}{\eta_D(\boldsymbol{\alpha})} \prod_{c=0}^1 \theta_c^{\alpha_c-1}$  with parameter vector  $\boldsymbol{\alpha} = (\alpha_0, \alpha_1)$  and  $\eta_D(\boldsymbol{\alpha}) = \Gamma(\alpha_0)\Gamma(\alpha_1)/\Gamma(\sum_c \alpha_c)$ . The expected value  $E(\boldsymbol{\theta}|\boldsymbol{\alpha})$  is  $\boldsymbol{\alpha}/\sum_c \alpha_c$ . If  $y$  has the counts  $\mathbf{m} = (m_0, m_1)$  with  $\sum_c m_c = n$ , we have the likelihood  $p(\mathbf{y}|\boldsymbol{\theta}) = \frac{1}{\eta_B(\mathbf{m})} \prod_{i=1}^n \theta_0^{y_i=0} \theta_1^{y_i=1}$  where  $\eta_B(\mathbf{m}) = \Gamma(m_0+1)\Gamma(m_1+$

$1)/\Gamma(n+1)$  and the expression  $y_i=c$  takes the value 1 if the observation  $y_i$  has value  $c$  and it takes the value 0 otherwise. Within leaf  $\mathcal{X}_\tau$  we get the posterior

$$p(\boldsymbol{\theta}|\mathbf{m}) = \frac{1}{\eta_{D\!i}(\mathbf{m} + \boldsymbol{\alpha})} \prod_{c=0}^1 \theta_c^{m_c + \alpha_c - 1} \quad (12)$$

which again is a Dirichlet density  $D\!i_2(\boldsymbol{\alpha} + \mathbf{m})$ .

A final point concerns the determination of the counts  $\mathbf{m} = (m_0, m_1)$  in a leaf of the tree. We have seen above that each region  $\mathcal{X}_\tau$  has various intersections with other regions  $\mathcal{X}_\eta$ , and that within these intersections the distribution is a weighted mixture of the binomial distributions with parameters  $\boldsymbol{\theta}_\tau$  and  $\boldsymbol{\theta}_\eta$  and known weights. Consequently the observations are attributed to the different regions with the same weights. For a given tree this uniquely determines the vector of counts  $\mathbf{m}_\tau$  relevant for a single region  $\mathcal{X}_\tau$ .

### 4.3 Comparing Splits with the Un-split Leaf

We want to assess the effect of splitting some leaf  $\mathcal{X}_\tau$  containing  $n_\tau$  observations into two subsets  $\mathcal{X}_{\tau_1}$  and  $\mathcal{X}_{\tau_2}$ . We assume that within  $\mathcal{X}_{\tau_i}$  the dependent variable is binomially distributed as  $Bi(\boldsymbol{\theta}_{\tau_i}, n_{\tau_i})$ . Hence we have to compare two hypotheses:

- $H_1$ : within each  $\mathcal{X}_\tau$ , the class variable  $\mathbf{y}$  has been generated according to a binomial distribution  $Bi(\boldsymbol{\theta}_\tau, n_\tau)$  with a single but unknown parameter  $\boldsymbol{\theta}_\tau$ .
- $H_2$ : within each  $\mathcal{X}_\tau$ , the class variable  $\mathbf{y}$  has been generated according to a binomial distribution  $Bi(\boldsymbol{\theta}_{\tau_i}, n_{\tau_i})$  with different unknown parameters  $\boldsymbol{\theta}_{\tau_i}$ .

We assume that on  $\mathcal{X}_\tau$  and  $\mathcal{X}_{\tau_i}$  the prior is Dirichlet  $p(\boldsymbol{\theta}|\boldsymbol{\alpha}_{\tau_i}) = D\!i(\boldsymbol{\alpha}_{\tau_i})$ . As the posterior again is Dirichlet we get after some tedious algebra the ratio of posterior probabilities of these hypotheses as

$$\frac{p(H_1|\mathbf{y}, \mathbf{X})}{p(H_2|\mathbf{y}, \mathbf{X})} = \frac{p(\mathbf{y}|\mathbf{X}, H_1)p(H_1)}{p(\mathbf{y}|\mathbf{X}, H_2)p(H_2)} = \frac{p(H_1) R_{D\!i}(\mathbf{m}_\tau, \boldsymbol{\alpha}_\tau)}{p(H_2) R_{D\!i}(\mathbf{m}_{\tau_1}, \boldsymbol{\alpha}_{\tau_1}) R_{D\!i}(\mathbf{m}_{\tau_2}, \boldsymbol{\alpha}_{\tau_2})}$$

where  $R_{D\!i}(\mathbf{m}, \boldsymbol{\alpha}) := \eta_{D\!i}(\mathbf{m} + \boldsymbol{\alpha})/\eta_{D\!i}(\boldsymbol{\alpha})$  and  $p(H_i)$  is the prior probability of a hypothesis. This is the *Bayes factor* for comparing models. Note that the probabilities  $\boldsymbol{\theta}_\tau$  within the leaves are integrated out and are not directly relevant for the selection of a split.

### 4.4 Possible Moves

Similar to [4] we make changes only at the bottom of the tree, as otherwise the calculation effort is larger

**GROW:** select a variable  $x_{s_\tau}$  for splitting and split a leaf  $\mathcal{X}_\tau$  into two new leaves  $\mathcal{X}_{\tau_1}$  and  $\mathcal{X}_{\tau_2}$ ,

**PRUNE:** collapse two leaves  $\mathcal{X}_{\tau_1}$  and  $\mathcal{X}_{\tau_2}$  into their common parent  $\mathcal{X}_\tau$ ,

**SHIFT:** move the split points  $\xi^+$  and  $\xi^-$  of the parent of two leaves,

**CHANGE:** split the parent of two leaves by another variable.

As each move has to cover its inverse we get three different move types: GROW/PRUNE, SHIFT and CHANGE. Obviously SHIFT does not change the number of parameters and is covered by the usual Hastings procedure. CHANGE is the combination of a PRUNE and a subsequent GROW. We demonstrate the algorithm for a GROW/PRUNE step. As the leaf probabilities  $\theta_\tau$  do not enter the model selection, the only parameters we have are the split index  $i$  of the variables to be split and the split points  $\xi^+$  and  $\xi^-$ .

We now have the task to define the different quantities in (9). The state  $(v, w)$  corresponds to a given tree, where  $w$  contains the continuous upper and lower split points for each nonterminal node. The  $m$ -th move is applicable only to two specific tree structures  $(v, w) := T_{m,1}$  and  $(\tilde{v}, \tilde{w}) := T_{m,2}$  and involves the split of a specific leaf  $\mathcal{X}_\tau$  of  $T_{m,1}$  yielding  $T_{m,2}$  or the pruning of children of  $\mathcal{X}_\tau$  in  $T_{m,2}$  again producing  $T_{m,1}$ . Note that for each different variable to be split we need a new move type.

$j_m(v, w)$  is the probability of selecting move  $m$  if the current states is  $(v, w)$ . We define it according to the following lines:

- The split variable is randomly selected with equal probability.
- In the initial phase GROW/PRUNE, SHIFT and CHANGE are selected with probabilities  $1/(3k)$ ,  $1/3$  and  $1/3$  respectively. This avoids that the tree grows too fast and a large number of split variables are not explored. After the tree has reached a stationary size, the probability of selecting GROW/PRUNE is increased. These probabilities are compensated and do not affect the limiting distribution, but only the convergence speed.
- Each eligible node (leaf to be split, or parent of two leaves to be pruned) is selected with identical probability.

The prior distribution  $p(v)$  is the prior distribution of different tree structures, i.e. trees of specific shapes. Our prior distribution depends on the number  $N$  of nodes in the tree. The prior probability of a tree is proportional to  $1/(1 + \beta \exp(\gamma N))$  with  $\beta, \gamma > 0$  and penalizes large trees. The variables used for splitting have equal prior probability to be selected.

As discussed by [3] it is legal and advantageous to let the prior depend on the data  $\mathbf{X}$  in some aspects. Assume  $x_{(1)} \leq x_{(2)} \leq \dots \leq x_{(l)}$  are the sorted observed values for some variable  $x_i$  in region  $\mathcal{X}_\tau$ . Then we first assume that both split points  $\xi^-$  and  $\xi^+$  are located in the interval  $(x_{(1)}, x_{(l)})$ . In addition we currently suppose that the overlap covers a fixed proportion  $\rho$  of the interval  $(x_{(1)}, x_{(l)})$ . Therefore  $\xi^-$  is has a uniform prior over  $(x_{(1)}, x_{max})$  with  $x_{max} = x_{(1)} + (1 - \rho)(x_{(l)} - x_{(1)})$ .  $\xi^+$  is defined as  $\xi^- + \rho(x_{(l)} - x_{(1)})$ .

This allows a simple definition of the quantities in (13). The auxiliary variable  $u$  is defined as univariate random variable with a uniform distribution  $p_m(u)$  in  $(x_{(1)}, x_{max})$  and  $\tilde{u}$  is empty. The map  $h_m(u, w)$  is just the identity, which leaves  $w$  unchanged. Hence the Jacobian determinant  $\left| \frac{\partial h_m(u, w)}{\partial (u, w)} \right|$  is equal to 1. Using these terms we may calculate the ratio of posteriors (13) defining the acceptance probability of a split.

## 4.5 Generating the Markov Chain

The procedure of generating the Markov chain involves the following steps

1. Randomly select a move  $m$  according to probability  $j_m(v, \mathbf{w})$ . In the case of GROW/PRUNE this involves
  - Determine whether two leaves are pruned or a split takes place, and which variable is to be split. In addition the leaf or parent node is determined.
  - If SPLIT is chosen, select a lower and an upper split point.
 For the other move types proceed analogously.
2. Calculate the ratio of posteriors (13).
3. Determine the acceptance probability by (9) and accept the new state or keep the old state.

The algorithm is iterated for some time until the number of nodes in the tree stabilizes. Then the resulting tree is saved and a new tree is grown. This yields a set  $\mathbf{w}_1, \dots, \mathbf{w}_B$  of parameters distributed according to the posterior distribution  $p(\mathbf{w}|y, \mathbf{X})$  and defines a representative ensemble of models. We then may estimate, for instance, the expected probability that a new object with features  $\mathbf{x}$  belongs to class  $c$  according to (2) by

$$E(q_c|\mathbf{x}, \mathbf{y}, \mathbf{X}) \approx \frac{1}{B} \sum_{i=1}^B p(y=c|\mathbf{x}, \mathbf{w}_b) \quad (13)$$

which converges to (2) by the Law of Large Numbers.

The work on Bayes trees is still in progress. So far we use fixed percentages of leaf overlap. This can be generalized to adaptively determine the optimal percentage of overlap for each pair of split points in each leaf of the tree. The approach may be readily generalized to more than 2 classes using multinomial distributions within the leaves. In contrast to usual tree approaches we need not determine the optimal probability for each subset of classes but may randomly select some subset. Therefore the computational effort should be reasonable.

## 5 Credit-Scoring and the Value of New Information

As a real-world application we selected the prediction of credit-worthiness of enterprises. Here  $\mathbf{x} = (x_1, \dots, x_k)$  are features of an enterprise and  $y$  indicate its *future solvency*:  $y=re$  if the enterprise is able to *repay* the credit and  $y=fa$  if it *fails* to repay. In the simplest case the bank has two options:  $a=de$ : “deny credit”, and  $a=gr$ : “grant credit”. In the numerical example of this paper we use the loss function with values  $L(a; y)$ :  $L(gr; re) = -9.6$ ,  $L(gr; fa) = 100$ ,  $L(de; re) = 0$ , and  $L(de; fa) = 0$ . We deny the loan if  $E(L(a=de)) < E(L(a=gr))$ .

Up to now the analysis usually centers on classifying a new credit applicant as solvent or insolvent [8]. But automatic credit-scoring is only the first step in a series of actions until a loan is granted. Therefore we investigate a scenario

where besides the usual decisions “grant credit” or “deny credit” there is a third option “collect additional evidence”, e.g. by a thorough audit of the enterprise. The last action will incur some additional cost  $c_{ev}$ . Subsequently we have to select a second action  $a_2$  and deny or grant the credit. We get the loss function

$$\begin{aligned} L(gr; re) &= -9.6; & L(gr; fa) &= 100; & L(de; re) &= 0; & L(de; fa) &= 0; \\ L(ev, gr; re) &= c_{ev} - 9.6; & L(ev, gr; fa) &= c_{ev} + 100; & L(ev, de; re) &= c_{ev}; & L(ev, de; fa) &= c_{ev}; \end{aligned}$$

$q_{re}$  was the probability that  $y$  has the value  $re$ . For a fixed  $x$  this probability value can be considered as a parameter with associated posterior density  $p(q_{re}|x, y, \mathbf{X})$ . To assess the effect of collecting additional evidence we need to model the information gain. The worst case is that we get no additional data. The best alternative is that *after* the collection of evidence we will know the *true* probability  $q_{re}$ . At the current stage we do not know the result of the investigation but only have the posterior distribution of  $q_{re}$ . Therefore it is consistent to assume that the ‘true’ probabilities will be distributed according to our current posterior density  $p(q_{re}|x, y, \mathbf{X})$ . So a realistic model may be a random mixture of both alternatives, say with mixture probability 0.5.

For simplicity we assume in this study that after the collection of evidence we will know the true probability  $q_{re}$ . Suppose action  $a_1 = ev$  is taken and afterwards we know  $q_{re}$ . Then we will take action  $a_2 = de$  if  $L(a_2 = de) < L(a_2 = gr)$ . By (3) this holds if  $q_{re} < \frac{L(ev, gr; fa) - L(ev, de; fa)}{L(ev, de; re) - L(ev, de; fa) - L(ev, gr; re) + L(ev, gr; fa)}$ . By averaging over the possible  $q_{re}$  we get the expected loss of decision  $a_1 = ev$

$$\begin{aligned} E(L(a=ev)|x, y, \mathbf{X}) &= & (14) \\ & \int_{p(q_{re}|x, w) > p_0} \sum_{y=re}^{fa} L(ev, gr; y) p(y|x, w) p(w|y, \mathbf{X}) dw \\ & + \int_{p(q_{re}|x, w) \leq p_0} \sum_{y=re}^{fa} L(ev, de; y) p(y|x, w) p(w|y, \mathbf{X}) dw \end{aligned}$$

We now select as  $a_1$  that action among  $gr$ ,  $de$ , and  $ev$  with minimal loss.

## 6 Results for Real World Data

We applied our procedure to a dataset containing 6667 records with 73 predictors. The predictors were drawn from balance sheet figures of 2263 enterprises. In the same way as [14] we divided the data randomly into a data set for training and model selection ( $n = 3701$ ) and a validation set ( $n = 2966$ ). To compare the tree methods with the procedures analyzed by [14] we determined a threshold on the validation set such that exactly 8.75% of the insolvent enterprises were wrongly accepted as solvent ( $\alpha$ -error) and subsequently measured the percentage of solvent enterprises which were rejected ( $\beta$ -error).

In his study [14] reports a  $\beta$ -error of 37.95% on the validation set for a pruned MLP as his best result, but remarks that this network had a  $\beta$ -error of 40.6% on the test set (which is part of our training set).

**Table 1.**  $\beta$ -error for different ensemble methods: Percentage of rejected solvent enterprises, if 8.75% of the insolvent enterprises were accepted as solvent.

| Algorithm                   | $\beta$ -error for ... |               |          |
|-----------------------------|------------------------|---------------|----------|
|                             | single model           | ensemble mean | CPU time |
| bootstrap LDA               | 69.54                  | 49.23         | 0.2 min  |
| bootstrap MARS              | 48.08                  | 39.90         | 90 min   |
| bootstrap CART              | 43.82                  | 36.62         | 15 min   |
| Bayes trees with no overlap | -                      | 38.66         | 30 min   |
| Bayes trees with overlap    | -                      | 33.72         | 90 min   |

For each of the algorithms CART (Classification And Regression Trees [2], LDA (linear discriminant analysis), MARS (Multivariate Adaptive Regression Splines [6]), and our Bayesian trees with and without overlap, we computed model ensembles of 500 members, using the data of the training set. The error percentages were computed using the data of the validation set. The results of our experiments are shown in table 1. In the column ‘single model’ there is the  $\beta$ -error of a single model estimated by maximum likelihood without resampling. ‘ensemble mean’ reports the  $\beta$ -error if all 500 ensemble members are used for prediction. The number of nodes of the trees varied from 100 to 300 with an average value of about 180. We used a fixed overlap of 20% determined by cross-validation. ‘CPU-time’ lists the approximate computing time for ensembles on a cluster of 14 Pentium PCs.

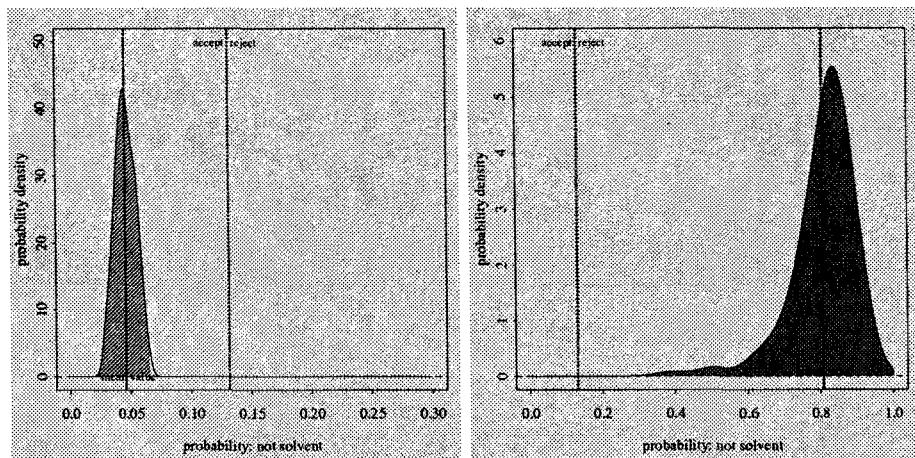
For trees with overlap the  $\beta$ -error on the validation set was consistently below 37% with a mean of about 33.7%. This shows that this procedure is able to beat MLPs in this classification task by about 5%. In addition the Bayesian trees have a small advantage over the bootstrap trees.

As in practical applications the  $\beta$ -error cannot be calibrated on the validation set. We used the value 0.9125 derived from the loss function as threshold. In this more realistic setting we got an  $\alpha$ -error of 7.5% and a  $\beta$ -error of 40.9% on the validation set.

Figure 1 shows different predictive densities for two different new inputs  $x$ . The decision boundary for rejection is shown as a vertical line near 0.13 and the mean value of the posterior distributions is a dotted line. The mean of the posterior distribution is far away from the decision boundary in both cases, and both distributions show low variance. Therefore the left case readily can be accepted and the right case can be rejected.

Figure 2 shows different predictive densities with high variances. They exhibit a large uncertainty on the right decision with a high probability of belonging to the ‘bad’ or the ‘good’ class. I.e. the mean posterior and the decision boundary are almost identical and there is a high variance. Obviously the prediction is unreliable in these cases, probably because the training data was sparse or contradictory in these input regions.

But there is a difference between the left and the right case: The right case also has a high variance due to insufficient comparable training data, but the



**Fig. 1.** Posterior density of the probability  $q_{fa}$  "no repay" for different new inputs. In these cases the decision is clear cut.

posterior probability mass is concentrated much further left, i.e. in the favorable area of solvent classification. Hence there is a larger chance to arrive at a prediction concentrated in the favorable area, if we get new information, e.g. add a few similar cases to the data base or acquire additional features of the case by a detailed investigation. Following this line of argument statistical decision theory allows to assess the value of new information. Taking into account cost and effort of a detailed investigation we can determine according to (15) whether it is reasonable to perform a detailed investigation.

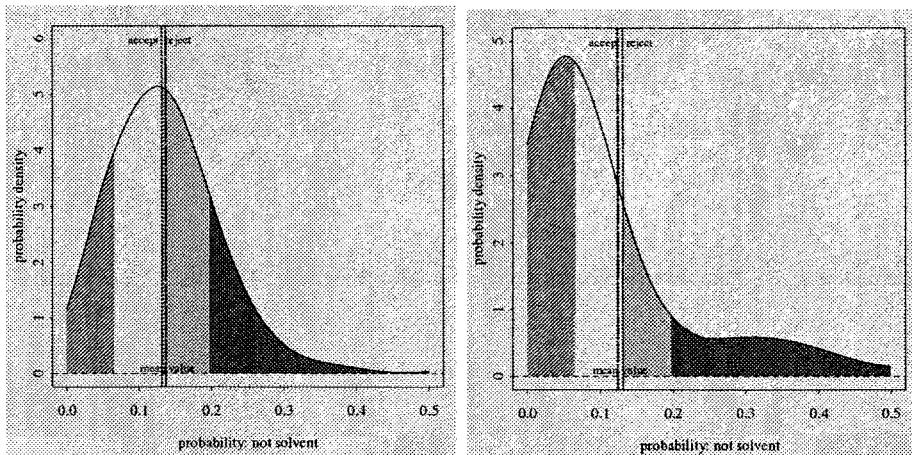
When we determined the decrease of loss by new evidence in our test set we got the following results. While for about 50% of the cases the decrease is negligible, about 20% of the cases have a decrease of more than 1.5. This is substantial as -9.6 was the maximum loss with a risk-free credit. Hence even if we assume that 1.5 is the cost of an investigation then more than 20% of the cases are rewarding for a closer screening.

### Acknowledgments

We thank Prof. Dr. Jörg Baetge, University of Münster for granting access to the dataset. The project was funded in part by the Real World Computing Partnership, Tsukuba, Japan.

### References

1. J.O. Berger. *Statistical Decision Theory, Foundations, Concepts and Methods*. Springer, New York, 1980.



**Fig. 2.** Posterior density of the probability  $q_{fa}$  "no repay" for different new inputs.

2. L. Breiman, J.H. Friedman, R. Olshen, and C.J. Stone. *Classification and Regression Trees*. Wadsworth International Group, Belmont, California, 1984.
3. W. Buntine. Learning classification trees. *Statistics and Computing*, 2:63–73, 1992.
4. H. Chipman, E. George, and R. McCulloch. Bayesian CART. Technical report, Department of Statistics, University of Texas at Austin, 1995.
5. L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer, New York, 1996.
6. J. H. Friedman. Multivariate adaptive regression splines. *Annals of Statistics*, 19(1):1–67, 1991.
7. J.H. Friedman. Local learning based on recursive covering. Technical report, Stanford Uni, August 1996.
8. J. Graf and G. Nakhaeizadeh. Recent development in solving the credit-scoring problem. In V.L. Plantamura, editor, *Logistic and Learning for Quality Software, Management and Manufacturing* ?, New York, 1993.
9. P. J. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. Technical report, Bristol Univ., 1995.
10. W.K. Hastings. Monte Carlo sampling methods using Markov chains and their application. *Biometrika*, 57:97–109, 1970.
11. G. Paass and J. Kindermann. Bayesian query construction for neural network models. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 443–450. The MIT Press, 1995.
12. B.D. Ripley. *Pattern Recognition and Neural Networks*. Cambridge University Press, Cambridge, 1996.
13. L. Tierney. Markov chains for exploring posterior distributions. Technical Report 560, School of Statistics, Univ. of Minnesota, 1994.
14. J. Wallrafen. Kreditwürdigkeitsprüfung von Unternehmen mit neuronalen Klassifikationsverfahren (credit scoring of enterprises with neural classification procedures). Master's thesis, University of Erlangen-Nürnberg, 1995.

# Contextual Text Representation for Unsupervised Knowledge Discovery in Texts

Patrick Perrin and Fred Petry

Department of Computer Science, Tulane University, New Orleans, LA 70118, U.S.A.

**Abstract.** This paper studies the role of lexical contextual relations for the problem of unsupervised knowledge discovery in full texts. Narrative texts have inherent structure dictated by language usage in generating them. We suggest that the relative distance of terms within a text gives sufficient information about its structure and its relevant content. Furthermore, this structure can be used to discover implicit knowledge embedded in the text, therefore serving as a good candidate to represent effectively the text content for knowledge elicitation tasks. We qualitatively demonstrate that a useful text structure and content can be systematically extracted by collocational lexical analysis without the need to encode any supplemental sources of knowledge. We present an algorithm that systematically extracts the most relevant facts in the texts and labels them by their overall theme, dictated by local contextual information. It exploits domain independent lexical frequencies and mutual information measures to find the relevant contextual units in the texts. We report results from experiments in a real-world textual database of psychiatric evaluation reports.

## 1 Introduction

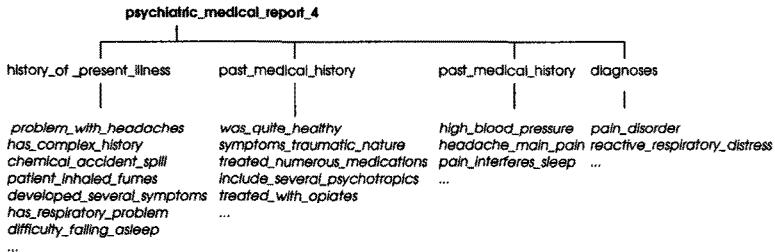
### 1.1 Knowledge discovery in full texts

Knowledge of language and domain knowledge are too expensive to build, and almost impossible to generate “on the fly” for any corpus, such as for a set of potentially interesting texts retrieved from the Internet. Therefore, we must find an easy and systematic way to impose a simple structure, yet one rich and descriptive enough to allow us to use discovery techniques and reveal interesting patterns in those texts. However, the main problem in dealing with unrestricted textual information is the ambiguity and infinite properties of natural languages (NL). An unrestricted text does not have a clear and finite set of features to be organized, say into the widely used vector representation. The number and type of the features are based and are dependent on the text content; therefore, they cannot be anticipated in advance. Features are text constructs related to the actual word usage in the text. Since the enumeration of all described phenomena to be encountered in any text is infinite, it is not possible to use it as the set of features. All these render difficult the mapping of textual information into manageable datasets in standard form. Despite tremendous progress in the natural language understanding (NLU) field of research, it is still very difficult

to extract the deep semantic structure of sentences found in any text, let alone the meaning of the text itself. Alternative solutions must then be developed and studied. Techniques and tools for the full automatic acquisition of useful information from textual databases are expected to be the most productive methods of building effective knowledge bases used for further complex cognitive tasks, such that problem solving, prediction, or information extraction. Knowledge discovery in texts (KDT) is the study of techniques for dealing with unrestricted textual patterns and their evaluation for interestingness. To reveal interesting patterns in a text, one must first impose some structure to the text before attempting any mining task. This paper discusses this aspect of KDT.

## 1.2 Approach

Finding the implicit significant structure of a text is a feature extraction problem [11]. Unrestricted declarative NL texts form the main category of texts found in technical textual databases (e.g., patient medical reports). These texts are scripted rather than spontaneous speech, have an implicit structure, and consist of the description of various features of the objects under consideration (e.g., a patient's case). Furthermore, they can be characterized by restricted lexical, syntactic, semantic, pragmatic and meta-textual properties. For example, there is little lexical ambiguity; the syntax is very regular; knowledge conveyed by the text is objective; metaphor is unlikely to occur; and overt structure (chapters, sections and so on) helps process long portions of text. Work in cognitive psychology and psycholinguistics suggest that (1) natural languages are so constructed that they are asymmetrically hierarchical; (2) meaning is mainly carried by groups of 2 to 4 words; (3) not all words are linguistically significant – content words convey the core of the meaning –; and (4) groups of content words are somewhat close to each other and usually span over more or less 5 words prior and after any word. Consequently, we define a textual pattern as 2 to 3 consecutive words within a loci of approximately 5 words. We call textual patterns collocational expressions to express the fact that they are groups of words related in meaning [9]. There exist various clues to find the natural (inherent) topic segmentation of a text. Mainly (1) by exploiting any explicit discourse structure, that is, by recognizing any pre-existing structure in the form of concise major sections headings (they usually partition the text into its various intended topics); and (2) by recognizing the fact that in declarative texts, words and constructs tend to be used with a consistent single intended meaning, and therefore the groups of sentences (paragraphs) containing mostly the same constructs, or words usage, are mostly likely to be about the same topics. They form then basic logical contextual units, which all together represent a linear segmentation of the text corresponding to its implicit discourse structure. In this inherent hierarchy of coherent contextual units, all concepts form intrinsic relations (strong associations). We referred to this by *text structure*, which we define as a sequence of major topics (called themes), each being a set of relevant assertions. This structure has then two major dimensions, which can be represented graphically. Figure 1 depicts a typical psychiatric evaluation report. Note that each assertion



**Fig. 1.** Example of a typical text structure

is a true statement and is relevant to its overall theme. This is due to the fact that all statements in the text are assumed to be true and valid. To be practical, both themes and relevant assertions must be extracted automatically from the text and no domain-dependent knowledge should be required. We propose to use a relative entropy measure, mutual information (MI) [14], to select the minimal set of collocational expressions representing all themes of the text.

### 1.3 Related work

Several algorithms have been developed to make use of information on lexical occurrence, such as to automatically assign part-of-speech tags to new sentences from their a-priori statistical occurrence in a large corpus [4], to recognize verb sub-categorization frames using syntactic local context [8], and for machine translation [3]. [5]'s approach to KDT is to study the corpus with respect to its concept distribution assuming that interesting discoveries will be hinted at by unexpected distributions. However, concepts are not automatically acquired from the corpus and it relies on tagged documents; and since manual tagging is expensive, very few databases are tagged. It is even unrealistic to consider that very large ever growing textual databases will be tagged ever [2]. [12] considered the problem of automatically generating extraction patterns for each noun phrase in untagged texts and requires the corpora to be somehow partitioned into relevant and irrelevant texts with respect to some specified subject of interest. [7] studied the problem of knowledge acquisition from texts as a top-down hierarchical clustering task. He reports that groups of pairs of words have been identified as relevant semantic contexts, when identified by their statistical occurrence in sentences. His goal is to learn a classifier that is able to recognize the discourse structure in the interviews of a car accident domain and to relate them and gain more knowledge about the domain. Finally, in a prior model, we used a fuzzy-based representation of the situation expressed by a text [10].

## 2 A model for contextual text processing

The objective is first, to find the implicit discourse topic structure in the text by identifying the major themes of the text. Second, to automatically extract and select the relevant collocational expressions in each theme.

## 2.1 Representing a text content

We used definite program clauses, a subset of a first-order logic, to represent the discourse topic structure and the relevant assertions. Each piece of information in a text is then a logical expression of the form:

$$\text{theme}(\text{collocational\_expression}, \text{text\_number})$$

which denotes that the particular concept stated by a collocational expression occurs in the text relevant to the stated case number, and that it has been observed to define the stated theme.

## 2.2 Discovering the discourse topic structure

We want to answer two questions: (1) what sort of structure is inherent in discourse, and (2) which mechanisms and aspect of language can be used to detect it. Certain aspects of the discourse structure can be revealed via lexical distribution patterns, discourse cues, and so on. The objective is to extract an overall topic structure to partition the text into non-fixed size contextual units, cohesive in their content, and therefore to assign a contextual label to whatever facts that will be extracted from such contextual units. This is a labeling problem and the proposed approach is simple, can be systematic and does not require an oracle, which are all desired qualities in a KDT paradigm. This makes the approach robust. For the class of technical texts, one can notice that the discourse is structured and often there is an explicit structure, in the form of major headings (e.g., *history\_of\_present\_illness*, *past\_medical\_history*). This structure is fairly easy to extract, requires no particular extensive knowledge encoding, and shows to be very useful to exploit. In the case there is no such explicit structure, Hearst proposed a straightforward method based on a contextual partitioning of the document [6]. Our objective in revealing the topic structure is to be able to cheaply label whatever is extracted from the identified sections. This is supplemental *local contextual information* that will prove useful for discovery of causal relationships. In the case of an explicit structure as in the psychiatric medical reports, whatever facts are extracted from the sections, say *history\_of\_present\_illness*, will be labeled as being about *history\_of\_present\_illness* (the theme predicate). Note that, by theme or topic, we only refer to signaling what pieces of information are generally about, depending on where they are extracted from the text. This is not to be confused with the topic of each particular sentence, which would require a deeper semantic analysis. Therefore, we only identify the intentional meaning (global interpretation) of clusters of conversational discourse [13].

## 2.3 Discovering significant text assertions

We developed an algorithm that automatically identifies prosodic segments (facts) in the text and evaluates them for their appropriateness. This approach is appealing because it provides a means to systematically take minimal measurements

from any unrestricted text, while ensuring the representation of all themes of the text. This is an unsupervised feature analysis problem where the collocational expressions are the measurements and the entropy measure serves at reducing the dimensionality of the space of features to render the discovery problem tractable. We augmented the traditional definition of collocates pairs of words to a more general definition due to the fact that co-occurrences of content words may not be adjacent [9]. A textual feature is a typical concept in a text unit, represented by a group of  $\alpha$  consecutive words which appear together within  $\beta$  words in a text unit. Features are identified by their relative frequency of occurrence (significance) in the text (context dependency). Since the definition of a textual feature is general enough to include any sequence of  $\alpha$  consecutive words, we used a measure of mutual information (MI) to distinguish the most significant ones, i.e., those bearing information which are most likely to help derive knowledge from the text. For example, with  $\alpha = 2$  and  $\beta = 5$ , all combinations of two consecutive words in the following sentence (respecting the word order in the sentence) are features:

*...during the examination, the patient denied any fever ...*

for example “*denied fever*”, “*examination patient*” or “*the patient*”, are features, but “*denied patient*” and “*during fever*” are not<sup>1</sup>. A collocational relation is a relevant textual feature, that is, a feature with high degree of intra-relation cohesion. As humans do discard irrelevant information in cognitive tasks, we model this aspect by ranking the extracted themes by their degree of relevancy in the text they describe. It has been found that relevance is relative to context, which renders the problem difficult because there is no way to know precisely which context someone has in mind at any given moment. It has been suggested that an assumption is relevant in a context to the extent that its contextual effects in this context are large; and it is irrelevant to the extent that the effort required to process it in this context is large [1]. Ordering collocational expressions by their information content reveals the most significant ones, which turns out to be those containing content words.

### 3 Results and discussion

We used a textual database of 557 psychiatric evaluation notes containing a total of 732968 words. Among these, 421254 are content words, with only 15390 different ones, suggesting a large repetition factor to describe psychiatric phenomena, even though the reports were generated by several different physicians. A psychiatric evaluation note is a medical report consisting of several pages of full narrative text transcribed from notes dictated during patient visits. The dataset corresponds to a mix of language usage and to various expertise on the same domain.

---

<sup>1</sup> “*denied patient*” does not follow the sentence word order and the distance between “*during*” and “*fever*” is more than 5 words.

### 3.1 Minimum theme coverage

Collocational expressions aim at describing the content of each contextual unit of a text. Therefore, the selected collocational expressions should discriminate the definition of each theme. This experiment has been designed to show that the use of the minimal MI has a discriminatory effect in selecting the collocational expressions to represent a text. At a first glance (table 1), and leaving alone their deep meaning, it seems that MI does just what we expected. It seems that the collocational expressions that were selected due to a high MI are actually good representative of each theme of the text. Each collocational expression is unique to a single theme; furthermore, there are no constituents of any of the collocational expressions appearing outside its theme. This suits the desirable discriminative text representation issue. Table 2 describes the minimum theme

**Table 1.** Minimum theme coverage for a typical report ( $\alpha = 3$  and  $\beta = 4$ )

| <i>history_of_present_illness</i> | <i>past_medical_history</i>  | <i>diagnosis</i>               |
|-----------------------------------|------------------------------|--------------------------------|
| coke_morphine_fentanyl            | gonorrhea_chlamydia_herpes   | narcissistic_borderline_traits |
| describe_waking_sweat             | harrington_rod_placement     |                                |
| downers_mentions_quaaludes        | harrington_rod_vertebral     |                                |
| hiv_test_unaware                  | mitral_valve_prolapse        |                                |
| morning_awakening_appetite        | rod_placement_vertebral      |                                |
| morphine_fentanyl_patches         | surgery_harrington_placement |                                |
| providing_emotional_support       | venereal_disease_gonorrhea   |                                |
| overtly_harming_mutilation        |                              |                                |
| quaaludes_coke_morphine           |                              |                                |
| quaaludes_coke_fentanyl           |                              |                                |
| retreating_sort_nonexistence      |                              |                                |
| receiving_elavil_making           |                              |                                |
| received_brief_detox              |                              |                                |
| shared_needles_knowledge          |                              |                                |

coverage for the text. It shows, for decreasing values of the mutual information content, how many collocational expressions cover each of the five themes the text contains. This table shows a minimum theme coverage for  $MI = 15$ , resulting in the minimal set of 38 selected collocational expressions. The minimal MI ensures an optimal selection of collocational expressions. If  $MI \ll MI_{mini}$  then more collocational expressions are selected, resulting that some of them will be present in more than one theme of the text and in a higher computational effort. A MI too high results in themes not being covered, and therefore, not being represented for the KD task. This is an undesirable loss of information. As one can see from table 2, the lower the MI, the more collocational expressions are

**Table 2.** Minimum theme coverage for a typical text

| <i>MI value</i> | <i>theme 4</i> | <i>theme 1</i> | <i>theme 3</i> | <i>theme 2</i> | <i>theme 8</i> | <i>total</i> |
|-----------------|----------------|----------------|----------------|----------------|----------------|--------------|
| 16              | 0              | 0              | 0              | 0              | 0              | 0            |
| 15              | 1              | 27             | 1              | 8              | 1              | 38           |
| 14              | 2              | 70             | 4              | 17             | 5              | 98           |
| 13              | 10             | 216            | 12             | 25             | 16             | 279          |
| 12              | 13             | 432            | 39             | 40             | 36             | 560          |
| 11              | 16             | 661            | 64             | 66             | 50             | 857          |
| 10              | 18             | 903            | 89             | 88             | 68             | 1166         |
| 9               | 22             | 1080           | 104            | 104            | 76             | 1386         |
| 8               | 22             | 1164           | 109            | 117            | 79             | 1491         |

selected. Therefore the goal to find the minimum MI for which all themes of the text are guaranteed to be represented by at least one collocational expression.

### 3.2 Collocational expressions “quality”

Table 3 is an example of the best 10 and the worse 10 collocational expressions for a typical report. First, one can notice that the frequencies of the selected collocational expressions are very low within a text. Even single word frequencies are quite low. For example, *patient* appears 73 times in the text. MI discriminates each theme by not selecting collocational expressions when the frequencies of their constituents are high with respect to the text. For example, the collocational expressions containing the constituent *patient* may not be good discriminatory ones, due to the fact that the word *patient* tends to be over represented in this report. That is, it is surely present in many, if not all, sections of the text. The collocational expressions best discriminating a text’s themes are those found in one or two occurrences in the whole text, that is, only in the theme. For example, the collocational expression *patient\_describes\_anxiety* in the second table could have been an interesting collocational expressions (from its meaning), but it was not selected because it was not informative, that is, although it was present only once in the text, its constituents were spread all over the text. These two tables clearly show that the “best” collocational expressions are actually those conveying the most informative, or interesting, message, from a human interpretation point of view. For example, the collocational expression *intended\_suicide\_events*, from the top of the table, is much more “interesting” than *was\_age\_was* from the bottom part of the table. We finally conducted an additional experiment to assess on the overall relevance of the set of selected collocational expressions for each text. The goal is to establish whether the selected collocational expressions are satisfactory in representing the text. This experiment involved 9 different persons (eliminating the individual bias) who

**Table 3.** Top and bottom 10 extracted collocational expressions ( $\alpha = 3, \beta = 4$ )

| <i>MI value</i> | $f(word_1)$ | $f(word_2)$ | $f(word_3)$ | $c_i$                          |
|-----------------|-------------|-------------|-------------|--------------------------------|
| 15.0759         | 1           | 1           | 1           | auditory_visual_hallucinations |
| 15.0759         | 1           | 1           | 1           | acts_rage_anger                |
| 15.0759         | 1           | 1           | 1           | describe_waking_sweat          |
| 15.0759         | 1           | 1           | 1           | hiv_test_unaware               |
| 15.0759         | 1           | 1           | 1           | hospital_admissions_psychiatry |
| 14.3828         | 2           | 1           | 1           | attempts_run_away              |
| 14.3828         | 1           | 1           | 2           | adolescence_dual_personality   |
| 14.3828         | 1           | 2           | 1           | brief_hospitalization_detox    |
| 14.3828         | 1           | 2           | 1           | disease_including_gonorrhea    |
| 14.3828         | 1           | 2           | 1           | intended_suicide_events        |
| 5.3873          | 73          | 13          | 17          | patient_denies_having          |
| 5.32699         | 34          | 28          | 18          | indicates_has_rather           |
| 5.24093         | 29          | 28          | 23          | states_has_not                 |
| 5.15068         | 8           | 73          | 35          | symptoms_patient_describes     |
| 5.03289         | 9           | 73          | 35          | currently_patient_describes    |
| 4.95652         | 73          | 34          | 10          | patient_indicates_had          |
| 4.88557         | 73          | 5           | 73          | patient_behavior_patient       |
| 4.69416         | 73          | 34          | 13          | patient_indicates_anxiety      |
| 4.09462         | 73          | 35          | 23          | patient_was_not                |
| 3.70376         | 73          | 34          | 35          | patient_indicates_was          |

were given randomly some of the 50 randomly selected texts from the whole corpus. They had to read the medical report, then to read the list of selected collocational expressions sorted by theme, and finally to make judgment whether the set is satisfactory or not in capturing the overall content of the report. From the 50 texts judged, 34 texts (68%) were considered satisfactory and 16 texts (32%) were not. One can fairly deduce from this qualitative measure the validation of the proposed approach. Recall that the objective were not to have the optimal axioms describing the text (not tractable) but to extract fairly good ones for KD tasks.

### 3.3 Role of the collocational expression parameters

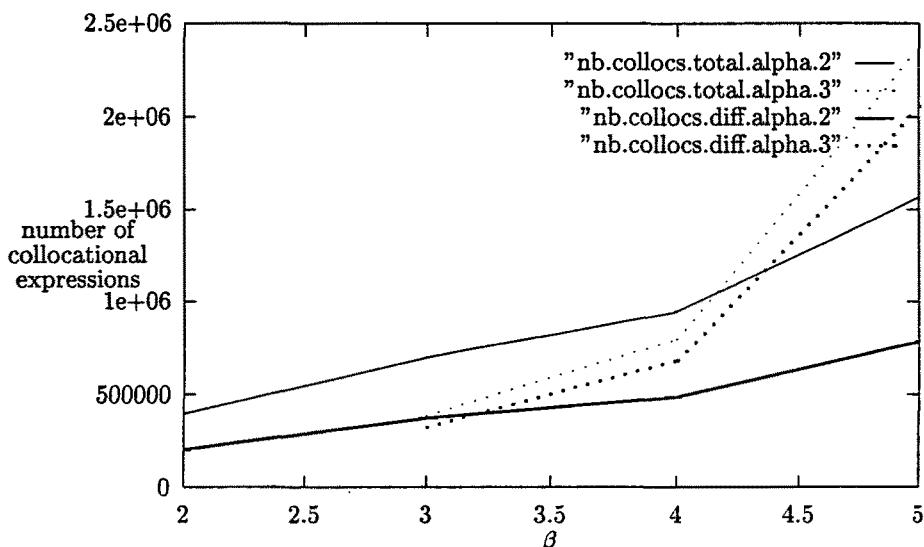
We aim at answering the following question: do  $\alpha$  and  $\beta$  play a role for contextual text processing? In order to study the extraction of relevant information by examining collocates in a given span, we compared various combinations of the loci length  $\beta = \{2, 3, 4, 5\}$  with the number of collocates  $\alpha = \{2, 3\}$ . These values correspond to the cognitive findings we mentioned earlier, evaluated to approximately 5 words (forward and backward) the span in which a collocate to

a given word will occur in a text. Considering more words for either the collocational expressions or for the loci does not make sense, linguistically speaking, since correlated words tend to be used close to each other's (short loci). Table 4 and figure 2 summarize some statistics in extracting collocational expressions for the various combinations. From this table, it is clear and predictable that

**Table 4.** Extraction of collocational expressions for various  $\alpha$  and  $\beta$

| $\alpha$ | $\beta$ | total collocations | different collocations | max frequency $H_{\alpha,\beta}(X)$ |
|----------|---------|--------------------|------------------------|-------------------------------------|
| 2        | 2       | 396988             | 202873                 | 1150                                |
| 2        | 3       | 699367             | 371483                 | 1136                                |
| 2        | 4       | 948400             | 488465                 | 1189                                |
| 2        | 5       | 1564748            | 783864                 | 1487                                |
| 3        | 3       | 387319             | 325230                 | 244                                 |
| 3        | 4       | 799851             | 684171                 | 283                                 |
| 3        | 5       | 2376222            | 2057282                | 312                                 |

the larger are  $\alpha$  and  $\beta$  the more collocational expressions are extracted. In order



**Fig. 2.** Extraction of collocational expressions for various  $\alpha$  and  $\beta$

to compare each combination  $\alpha, \beta$ , we computed the entropy  $H_{\alpha,\beta}(X)$  assuming that all collocational expressions are values of a single discrete random variable  $X$ . Recall that the entropy is given by:

$$H_{\alpha,\beta}(X) = - \sum_{i=1}^n p_i \log_2(p_i)$$

where  $n$  is the total number of collocational expressions and  $p_i$  is the probability that  $X$  is the collocational expression  $i$ , which is simply defined by its frequency of occurrence in the corpus. Table 5 summarizes the entropies for combinations of  $\alpha$  and  $\beta$ . The idea is to evaluate the mean amount of information for each

**Table 5.** Entropy for various  $\alpha$  and  $\beta$

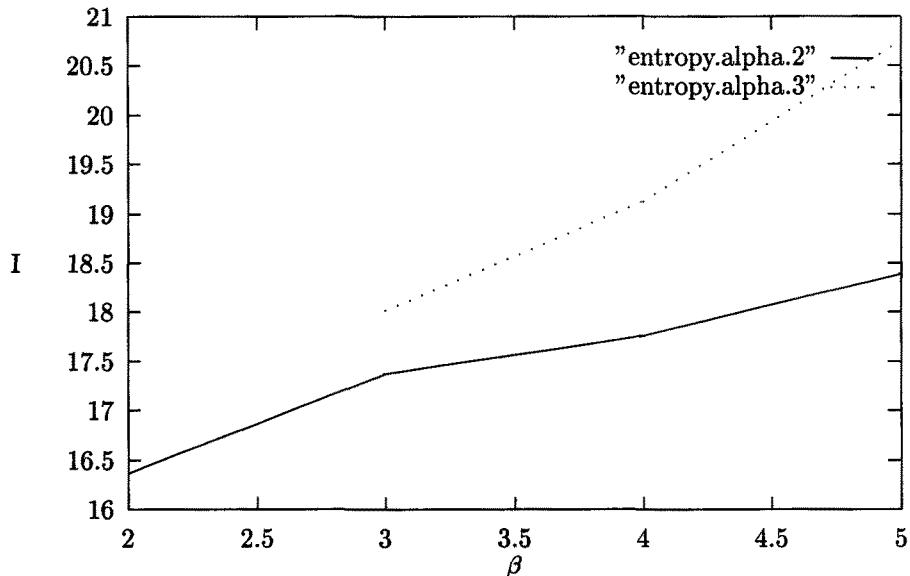
| $\alpha$ | $\beta$ | total collocations | $H_{\alpha,\beta}(X)$ |
|----------|---------|--------------------|-----------------------|
| 2        | 2       | 396988             | 16.369                |
| 2        | 3       | 1564748            | 17.377                |
| 2        | 4       | 1564748            | 17.7574               |
| 2        | 5       | 1564748            | 18.3956               |
| 3        | 3       | 799851             | 18.0172               |
| 3        | 4       | 799851             | 19.128                |
| 3        | 5       | 1484782            | 20.7503               |

combination and decide, with respect to this measurement, which one is more advantageous. The results suggest that the highest  $\alpha$  and the highest  $\beta$  then the highest the gain of information. As expected, the combination  $\alpha = 3$  and  $\beta = 5$  seems to give the best results, with respect to the representation of the content of the corpus. The drawbacks are the larger number of collocations and therefore a higher computational effort (time).

## 4 Conclusions

We demonstrated that a useful text structure and content can be systematically extracted by collocational lexical analysis without the need of any extra sources of knowledge. We have described algorithms to model autonomous extraction of relevant facts from a text and assign each with a theme label obtained from the automatic extraction of the topic structure discourse. The advantages of this approach are:

- Statistical methods to extract and sort by MI collocational expressions are very attractive, mainly because they are simple and are domain independent – they only depend on the actual content of a text.



**Fig. 3.** Entropy for various  $\alpha$  and  $\beta$

- There is no deep parsing of the texts, which reduces considerably the complexity of utilizing such information because of the fact that the texts are narrative and therefore contains jargon, non-well-formed sentences, etc. which is a real challenge for actual NLP grammars and unrestricted text parsers.

The limitations are that the proposed text content extraction algorithm is blind with respect to the semantics of the word correspondences it makes. It only gives an abstract result of the most relevant word associations a human reader would outline from the text, although these associations reflect the semantic context of NL.

We have shown then an effective and “cheap” method to extract facts reflecting the content of the text and their automatic labeling reflecting their context without the need of an oracle or any manually constructed training sets, but only based on the text discourse structure and contextual information. It works for the class of text we have considered and is best suited for the intended use of this information, such as knowledge discovery tasks. This may not be suitable for tasks requiring deeper understanding using the actual meaning of each phrasal component. This approach, despite possible imperfection, is still more efficient than extracting facts by hand and more robust than attempting to parse large sets of any long texts. These are necessary conditions for KDD techniques to be practical in real world applications. Finally, we reported results on the psychiatric database which have demonstrated the qualitative validation of the model.

## 5 Acknowledgments

Thanks to Marcos Fe-Bornstein, M.D. at Tulane Medical Hospital for providing valuable datasets of psychiatric evaluation reports.

## References

1. Varol Akman and Mehmet Surav. Steps toward formalizing context. *AI Magazine*, 17(3):55–72, Fall 1996.
2. Eric Brill and Raymond J. Mooney. An overview of empirical natural language processing. *AI Magazine*, 18(4):13–24, Winter 1997.
3. P. Brown and J. Cocke *et al.* A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85, jun 1990.
4. Kenneth Church, William Gale, Patrick Hanks, and Donald Hindle. Using statistics in lexical analysis. In U. Zernick, editor, *Lexical acquisition: exploiting on-line resources to build a lexicon*, chapter 6, pages 115–164. LEA, Hillsdale, NJ, 1991.
5. Ronen Feldman and Ido Dagan. Knowledge discovery in textual databases (kdt). In U. Fayyad and R. Uthurusany, editors, *Proceedings of the First International Conference on Knowledge Discovery and Data Mining (KDD-95)*, pages 112–117, 1995.
6. Marti A. Hearst. *Context and structure in automated full-text information access*. PhD thesis, Computer Science Division, University of California at Berkeley, 1994.
7. Stephane Lapalut. Text clustering to support knowledge acquisition from documents. Research Report 2639, INRIA, aug 1995.
8. Christopher Manning. Automatic acquisition of large subcategorization dictionary from corpora. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, pages 235–242, 1993.
9. Patrick Perrin. *Contextual Representation and Learning for Unsupervised Knowledge Discovery in Texts*. PhD thesis, Computer Science Department, Tulane University, New Orleans, LA, March 1997.
10. Patrick Perrin and Fred Petry. Fuzzy feature analysis for unsupervised knowledge discovery in narrative texts. In *Proceedings of the 6th International Conference on Fuzzy Systems (FUZZ-IEEE'97), Barcelona, Spain*. IEEE, jul 1997.
11. Patrick Perrin and Fred Petry. On lexical contextual relations for the unsupervised discovery of texts features. In H. Liu and H. Motoda, editors, *Feature Extraction, Construction, and Selection - A Data Mining Perspective*. Kluwer Academic, 1998. (to appear).
12. Ellen Riloff. Automatically generating extraction patterns from untagged text. In AAAI Press/MIT Press, editor, *Proceedings of the 13th National Conference on Artificial Intelligence*, pages 1044–1049, Menlo Park, CA, 1996.
13. John Rotondo. Clustering analysis of subjective partitions of text. *Discourse Processes*, 7:69–88, 1984.
14. Claude Shannon and Warren Weaver. *The mathematical theory of communication*. University of Illinois Press, Urbana, IL, 1963.

# Treatment of Missing Values for Association Rules

Arnaud Ragel and Bruno Crémilleux

ragel@info.unicaen.fr – cremilleux@info.unicaen.fr  
GREYC, CNRS UPRESA 6072  
Université de Caen  
F 14032 Caen Cedex France

**Abstract.** Agrawal *et al.* [2] have proposed a fast algorithm to explore very large transaction databases with association rules [1]. In many real world applications data are managed in relational databases where missing values are often inevitable. We will show, in this case, that association rules algorithms give bad results because they have been developed for transaction databases where practically the problem of missing values does not exist. In this paper, we propose a new approach to mine association rules in relational databases containing missing values. The main idea is to cut a database in several valid databases (vdb) for a rule, a vdb must not have any missing values. We redefine *support* and *confidence* of rules for vdb. These definitions are fully compatible with [2] which is the core of all known algorithms. Simulations show that this approach outperforms the classic approach by factors five to ten.

**keywords:** Association rules, Missing values, Knowledge Discovery.

## 1 Introduction

Data mining (knowledge discovery in databases) is a field of increasing interest combining databases, artificial intelligence and machine learning. The purpose of data mining is to facilitate understanding large amounts of data by discovering regularities or exceptions.

Discovery of association rules [1] is an interesting subfield of data mining. Historically, the motivation for searching association rules has come from the desire to analyze large amounts of supermarket basket data. For instance, an association rule *beer*  $\rightarrow$  *chips*(87)% states that 87% of customers that have bought beer, also have bought chips. More generally, given a transaction database, where each data is a set of literals (called items), an association rule is an expression of the form  $X \rightarrow Y$ , where X and Y are sets of items. The intuitive meaning of such a rule is that data of the database which contains X tends to contain Y. A rule  $X \rightarrow Y$  is evaluated by a support and a confidence. Its support is the percentage of data that contains both X and Y. Its confidence is the percentage of data that contains Y among those containing X. The problem of mining association rule is to find all rules that satisfy a user-specified minimum support and minimum confidence. Since Agrawal *et al.* [1], association rules have been the

focus of considerable work on developing fast algorithms, e.g [2] [16] [15] [10]. At the present time, they are used to explore millions of data [4].

Missing values have not been considered of interest since [1] [2] because association rules have been first developed to explore transaction databases where the problem of missing attribute values does not practically exist. However this problem becomes important if we try to find associations between values of different attributes in relational tables where missing values are often inevitable. Consequently all the algorithms derivated from [1] are ineffective in this context: very few rules are discovered in databases containing missing values (in Sect. 5.1 we will show the slump of the results). In this paper we present an extension for [2], which is the core of all known algorithms, to discover association rules in databases with missing attribute values. This approach is inspired by one of the simplest way to treat missing values in machine learning methods. It consists in evaluating a rule only with known values, i.e ignoring missing values. Even though this approach causes a huge waste of data for a lot of methods, e.g decision trees [9] [14], it is particularly well suited for association rules: a cancelled data for a rule, i.e data containing missing values for values tested by the rule, can be used by others rules. Then, even if the whole database is not used to evaluate a rule, the whole database is used to discover the ruleset. So a database is divided in several valid databases, where for a rule, a vdb must not have any missing values. Support and confidence must have been redefined to be evaluated in the different valid databases.

The remaining part of the paper is organized as follows. Section 2 briefly reviews the definition of association rules and the core of efficient algorithms. In Sect. 3, we will begin with a short presentation of the missing values and next we will set out problems posed by missing values for association rules. In Sect. 4 we will present our solution. In Sect. 5, we will present a simulation where we introduce, artificially, missing values to show the slump of the results with the usual approach and the robustness of the new one. Results on real world experiment (medical databases on Hodgkin disease) are presented too. Finally, we will conclude in Sect. 6 where we show how our extension can be useful to fill missing values for machine learning methods like decision trees.

## 2 Association Rules

We present a definition of association rules for relational databases to focus on missing values. Actually, association rules can be used on more general datasets (see [1]).

### 2.1 General Definitions

In databases, data are described by a same set of attributes, where each of them have a value which is chosen in a set of possible values, specific to each attribute. For example, in Fig. 1 (Section 3.2) the database has 4 attributes ( $X_1, X_2, X_3, X_4$ ) and the possible values are a,b for  $X_1$ , a,b,c for  $X_2$ , a,b,c,d,e,f,g,h for  $X_3$  and c,d for  $X_4$ .

Let  $B$  be a database with  $R = \{I_1, I_2, \dots, I_m\}$ , the set of all associations attribute values, called items. A data  $t$  is composed by a subset of  $R$  ( $t$  is a tuple). For the precedent example, the set of all the items is  $\{X1=a, X1=b, X2=a, X2=b, X2=c, X3=a, \dots, X3=h, X4=c, X4=d\}$

An association rule is an implication of the form  $X \rightarrow Y$  with  $X \subseteq R$ ,  $Y \subseteq R$  and  $X \cap Y = \emptyset$ . A rule is evaluated by a support and a confidence where it has a support  $s$  in the database if  $s\%$  of data contain  $XY$  and a confidence  $c$  if  $c\%$  of data that contain  $X$  also contain  $Y$ .

**Definition 1 (Support).** *The support of an itemset  $X \subseteq R$  in a database  $B$  is:*

$$\text{Support}(X) = \frac{|\{t \in B \mid X \subseteq t\}|}{|B|} \quad (1)$$

*The support of a rule  $X \rightarrow Y$  in a database  $B$  is:*

$$\text{Support}(X \rightarrow Y) = \text{Support}(XY) \quad (2)$$

**Definition 2 (Confidence).** *The confidence of a rule  $X \rightarrow Y$  in a database  $B$  is:*

$$\text{Confidence}(X \rightarrow Y) = \frac{\text{Support}(XY)}{\text{Support}(X)} \quad (3)$$

*Remark 1.* In the remaining of the paper we note, for an itemset  $X \subseteq R$ ,  $B_X$  the subset of a database  $B$  containing  $X$ , i.e  $B_X = \{t \in B \mid X \subseteq t\}$

The problem of mining association rules is, given a database  $B$ , to generate all association rules that have support and confidence greater than user-specified minimum support and minimum confidence respectively.

A confidence threshold is used to exclude rules that are not strong enough to be interesting. A support threshold is also given to remove rules that do not apply often enough.

The problem of mining association rules can be decomposed into two sub-problems [1]:

- Find all combinations of items that have data support above minimum support. Call those combinations frequent itemsets.
- Use the frequent itemsets to generate the desired rules. The general idea is that if  $ABCD$  and  $ABD$  are frequent itemsets, then we can determine if a rule  $ABD \rightarrow C$  holds by computing the ratio  $r = \text{support}(ABCD)/\text{support}(ABD)$ . The rule holds only if  $r$  is superior to minimum confidence.

The first step is responsible for most of the computation time (the second step just consists in a division of support between frequent itemsets).

An efficient breadth-first, or level-wise method for generating candidate sets, i.e potentially frequent itemsets, has been presented in [2] [3]. This method, called Apriori, is the core of all known algorithms.

## 2.2 Breadth-first Algorithms

The main idea is that if a set is not frequent, i.e with support lower than the user specified support, then its supersets can not be frequent. The algorithm starts on level 1 by evaluating the support of singleton itemsets. On level  $k$ , candidate itemsets  $X$  of size  $k$  are generated such that all subsets of  $X$  are frequent. After the support of the candidates on level  $k$  have been evaluated, new candidates for level  $k+1$  are generated and evaluated. This is repeated until no new candidates can be generated. The efficiency of this approach is based on not generating and evaluating those candidate itemsets that can not be frequent, given all smaller frequent itemsets. Further more, the whole database is not used for evaluating the support of candidate itemsets after the first pass because each itemset matches a subset of data: in later passes, the size of the subset can become much smaller than the database, thus saving much reading effort.

As an example, let  $L_3$  be  $\{\{1\ 2\ 3\}, \{1\ 2\ 4\}, \{1\ 3\ 4\}, \{1\ 3\ 5\}, \{2\ 3\ 4\}\}$ <sup>1</sup>, the frequent itemsets of size 3. Only  $\{1\ 2\ 3\ 4\}$  is a possible itemset of length 4 (in  $\{1\ 3\ 4\ 5\}$ , the itemset  $\{1\ 4\ 5\}$  is not frequent, i.e is not in  $L_3$ ). The data matched by this itemset are the intersection between the data matched by  $\{1\ 2\ 3\}$  and  $\{1\ 2\ 4\}$ .

In the next section the problem posed by the missing values is studied.

## 3 Association Rules and Data with Missing Values

### 3.1 Data and Missing Values

The problem of the missing values has been widely studied in the literature [9] [12] [6]. A missing value hides a value of an attribute. There are various way in which missing values might occur. For example:

- values recorded are missing because they were too small or too large to be measured.
- values recorded are missing because they have been forgotten, they have been lost (input error) or they were not available.

In the first case the missing values are structured, i.e they hide a same category of values for an attribute. In the previous example they only affect large and small values. In the second case values are missing at random.

Now, let's show the weakness of the usual treatment of association rules when data contain missing values.

### 3.2 Usual Treatments

Association rules have been first developed to explore transaction databases where the problem of missing attribute values does not practically exist. This

---

<sup>1</sup> each number represent an attribute-value association

problem becomes important if we try to find associations between values of different attributes in relational tables. To our knowledge, no studies and no applications, to deal efficiently with missing values, have been made since. Consequently and actually, missing values are not correctly treated. In effect missing values are treated like a new possible value for each attribute. This is equivalent to the method presented in [11] for missing values in decision trees which is only appropriate when the missing values are informative, e.g. values recorded are missing because they were too small or too large to be measured [7] [8].

Unfortunately, values are often missing at random. In this case the value  $?$  does not have the same status as a proper attribute value because, for a same attribute,  $?$  is not equivalent between data, i.e. the  $?$  hides several different values. The result of such a treatment, in this case, is very bad.

Example:

Imagine an attribute A with two possible values,  $a_1$  and  $a_2$ , with  $t_{a_1}\%$  for  $a_1$  and  $t_{a_2}\%$   $a_2$  ( $t_{a_1} + t_{a_2} = 100\%$ ). The item  $A = a_1$  (resp.  $A = a_2$ ) will have a support of  $t_{a_1}\%$  (resp.  $t_{a_2}\%$ ). If missing values are introduced with  $vm_{a_1}\%$  for the value  $a_1$  and  $vm_{a_2}\%$  for the value  $a_2$ , the  $A = a_1$  (resp.  $A = a_2$ ) will have a support of  $t_{a_1} - vm_{a_1}\%$  (resp.  $t_{a_2} - vm_{a_2}\%$ ). Further more the item  $A = ?$  will have a support  $vm_{a_1} + vm_{a_2}\%$ .

This solution reduces support and confidence values. With the combination of items in an itemset, supports are reduced a lot and, so, can be under the user specified support. The result is a modification of the rule evaluations (support and confidence) or a loss of rules. In Sect. 5.1 a simulation with several databases shows that with 15% of missing values, more than 90% of the rules are lost.

We give in Fig. 1 a very simple example which puts the problem to the fore quite well. We give two similar databases with 4 attributes: X1, X2, X3 and X4. In the second table we have introduced artificial random missing values.

| Id               | X1 | X2 | X3 | X4 | Id               | X1 | X2 | X3 | X4 |
|------------------|----|----|----|----|------------------|----|----|----|----|
| 1                | a  | a  | a  | c  | 1                | ?  | a  | a  | c  |
| 2                | a  | a  | b  | c  | 2                | a  | a  | b  | ?  |
| 3                | a  | b  | c  | c  | 3                | a  | b  | c  | c  |
| 4                | a  | b  | d  | c  | 4                | a  | b  | d  | c  |
| 5                | b  | b  | e  | d  | 5                | ?  | b  | e  | d  |
| 6                | b  | b  | f  | d  | 6                | b  | b  | f  | ?  |
| 7                | b  | c  | g  | d  | 7                | b  | c  | g  | d  |
| 8                | b  | c  | h  | d  | 8                | b  | c  | h  | d  |
| Database 1 (DB1) |    |    |    |    | Database 2 (DB2) |    |    |    |    |

Fig. 1. Same database, without and with missing values

If a minimum support of 40% and a minimum confidence of 80% are chosen we have, from DB1:

Itemsets with a support of 50%:

length 1: {X1=a}, {X1=b}, {X2=b}, {X4=c}, {X4=d}

length 2 (constructed from itemsets of length 1): {X1=a, X4=c}, {X1=b, X4=c}

As a result we have 4 rules:

If X1=a → X4=c, support=50, confidence=100

If X1=b → X4=d, support=50, confidence=100

If X4=c → X1=a, support=50, confidence=100

If X4=d → X1=b, support=50, confidence=100

In the second database, containing missing values, all the rules are lost because supports of the itemsets are under the minimum support of 40% (except for X2=b). If we look for the values of the precedent rules in DB2 we found a support of 25% and a confidence of 66%. If we choose a new minimum support of 25% and a new confidence support of 66% we retrieve the precedent rules but with a lot of new rules:

If X1=a → X4=c, support=25, confidence=66,7

If X1=b → X4=d, support=25, confidence=66,7

If X4=c → X1=a, support=25, confidence=66,7

If X4=d → X1=b, support=25, confidence=66,7

Here are the new rules:

If X2=c → X1=b, support=25, confidence=100

If X2=c et X4=d → X1=b, support=25, confidence=100

If X4=c → X2=b, support=25, confidence=66,7

If X1=a → X2=b, support=25, confidence=66,7

If X1=a et X4=c → X2=b, support=25, confidence=100

If X4=d → X2=c, support=25, confidence=66,7

If X1=b → X2=c, support=25, confidence=66,7

If X1=b et X4=d → X2=c, support=25, confidence=100

If X2=b et X4=c → X1=a, support=25, confidence=100

If X2=c → X1=b, support=25, confidence=100

If X1=a et X2=b → X4=c, support=25, confidence=100

If X2=c → X4=d, support=25, confidence=100

If X1=b et X2=c → X4=d, support=25, confidence=100

To sum up, to discover useful rules in a database containing missing values, support and confidence could be reduced. It is a first problem because it is a hard problem to fix these minimum values. A second problem is that with these new thresholds, the useful rules are under evaluated and they are hidden by a lot of new inconsistent rules. In this example we know that only the four first rules are useful (because we have DB1 without missing values) but in real situations it is impossible to make a difference between useful and inconsistent rules when useful rules are under evaluated by the presence of missing values. Consequently association rules are not of interest in domains with missing values.

In the next section, we present our new approach where we avoid these precedent problems by partially ignoring missing values. With this new approach the

rules are not evaluated by the presence of missing values. Consequently rules can be used just as if there were not any missing values.

## 4 New Approach to Deal with Missing Values

As we do not know what random missing values mean (a ? hides, for a same attribute, different values for each data) we will ignore them. In fact only certain data containing missing values will be partially disabled.

### 4.1 Definitions

Let  $B$  be a database,  $X$  an itemset,  $t$  a data and  $R$  a rule  $X \rightarrow Y$ .

**Definition 3 (Disabled data).**  $t$  is disabled for  $X$  in  $B$ , if  $t$  contains missing values for at least one item  $i$  of  $X$ .

**Definition 4.** We note  $Dis(X)$  the subset of  $B$  disabled for  $X$ .

**Definition 5.** We note  $B_X$  the subset of  $B$  containing  $X$ , i.e  
 $B_X = \{t \in B \mid X \subseteq t\}$

**Definition 6 (Valid database).** The valid database (vdb) for an itemset  $X$  in  $B$  is:  $vdb(X) = B - Dis(X)$

The valid database (vdb) for a rule  $R$  in  $B$  is:  $vdb(R) = B - Dis(XY)$

In case of random missing values,  $vdb(X)$  is a sample without missing values of  $B$ . Consequently we can calculate the support of an itemset in its vdb and retrieved the support as if the database did not contain missing values. With this approach the whole database is not used to evaluate an itemset but the whole database is used to evaluate all itemsets because each itemset has a different vdb.

Example of disabled data:

Data 1:  $X1=a$ ,  $X2=?$ ,  $X3=a$ .

Data 1 is disabled for the itemset  $\{X1=a \& X2=c\}$  because we do not know if the ? hide a  $c$  or another value. If the real value is  $c$  the support must be increased but if it is another value the support must be decreased. As we can not conclude, the data is disabled. On the contrary data 1 is not disabled for the itemset  $\{X1=a \& X3=a\}$ .

Data 1 is disabled for an itemset like  $\{X1=b \& X2=c\}$  even if, whatever the value hidden by ?, this itemset can not verify it. In this way, a vdb for an itemset  $Z$ , can be a good sample.

The main problem is now to calculate efficiently support and confidence in vdb for rules. In the next section we give new support and confidence definitions, fully compatible with [2], and we introduce a new parameter called *representativity*.

## 4.2 Support, Confidence and Representativity in Valid Databases

**Notation:**  $|B| = Card(B)$

**Support:** We redefine support to calculate it ignoring disabled data.

$$Support(X) = \frac{|B_X|}{|B|} \quad (\text{usual definition}) \quad (4)$$

$$Support(X) = \frac{|B_X|}{|vdb(X)|} = \frac{|B_X|}{|B| - |Dis(X)|} \quad (\text{new definition}) \quad (5)$$

Figure 2 gives results from databases of the Fig. 1 for the same parameters of the previous example (minimum support of 40% and minimum confidence of 80%). We retrieved the results of the Sect. 3.2. No itemsets are erroneously discovered and they are not under evaluated.

| Itemsets   | data disabled | support |
|------------|---------------|---------|
| X1=a       | 1, 5          | 3/6=50% |
| X1=b       | 1, 5          | 3/6=50% |
| X2=b       |               | 4/5=50% |
| X4==c      | 2, 6          | 3/6=50% |
| X4=d       | 2, 6          | 3/6=50% |
| X1=a, X4=c | 1, 2, 5, 6    | 2/4=50% |
| X1=b, X4=d | 3, 4, 7, 8    | 2/4=50% |

Fig. 2. Correction of supports for missing values

**Confidence:** A rule  $X \rightarrow Y$  holds with confidence of c% of data that contain X also contain Y.

$$Confidence(X \rightarrow Y) = \frac{|B_{XY}|}{|B_X|} \quad (\text{usual definition}) \quad (6)$$

Confidence is quick and easy to calculate because  $|B_X|$  and  $|B_{XY}|$  have already been calculated in  $Support(X)$  and  $Support(XY)$ . In effect:

$$\frac{Support(XY)}{Support(X)} = \frac{\frac{|B_{XY}|}{|B|}}{\frac{|B_X|}{|B|}} = \frac{|B_{XY}|}{|B_X|} \quad (7)$$

With the new approach it is not so easy. In a rule  $X \rightarrow Y$ , a data D, can be not disabled for X and disabled for XY, i.e  $B_X$  can contain disabled data for XY. Then the ratio  $\frac{|B_{XY}|}{|B_X|}$  is falsified by missing values of Y. We must

recalculate the support of X in the valid database of XY. In fact we can use previous calculus:

$$\text{Confidence}(X \rightarrow Y) = \frac{|B_{XY}|}{|B_X| - |Dis(Y) \cap B_X|} \quad (\text{new definition}) \quad (8)$$

$|B_{XY}|$  has already been calculated in  $\text{Support}(XY)$ ,  
 $|B_X|$  has already been calculated in  $\text{Support}(X)$ ,  
 $|Dis(Y) \cap B_X|$  requires to scan  $Dis(Y)$ , which has already been constructed during the calculation of  $\text{Support}(Y)$ , and to keep those which contain X.

Figure 3 shows the result for the database of the Fig. 1, containing missing values, according to a minimum confidence of 80%. With our new definitions, the presence of missing values is well suited. Exactly the same ruleset is retrieved where as with the usual approach all the rules were lost.

| rule $X \rightarrow Y$      | data disabled<br>for X | data disabled<br>for Y | support  | confidence    |
|-----------------------------|------------------------|------------------------|----------|---------------|
| $X1 = a \rightarrow X4 = c$ | 1,5                    | 2,6                    | $2/4=50$ | $2/(4-2)=100$ |
| $X1 = b \rightarrow X4 = d$ | 1,5                    | 2,6                    | $2/4=50$ | $2/(4-2)=100$ |
| $X4 = c \rightarrow X1 = a$ | 2,6                    | 1,5                    | $2/4=50$ | $2/(4-2)=100$ |
| $X4 = d \rightarrow X1 = d$ | 2,6                    | 1,5                    | $2/4=50$ | $2/(4-2)=100$ |

Fig. 3. Correction of confidence for missing values

**Representativity:** To obtain good results a vdb must be a good sample of the database. This is normally true if values are missing at random. To avoid rules to be discovered in a small vdb (over specified rules) we introduce a new parameter called *Representativity*.

$$\text{Representativity}(X) = \frac{|vdb(X)|}{|B|} = \frac{|B| - |Dis(X)|}{|B|}$$

To be usable an itemset X must have a representativity greater than a user-specified minimum representativity.

#### 4.3 Comments

These new definitions are fully compatible with [2] and quick to calculate: like in [2] where  $B_{XY} = B_X \cap B_Y$ , we have  $Dis(XY) = Dis(X) \cup Dis(Y)$ .

We have seen that our approach is well suited to cope with random missing values. We show now that it always deals with informative missing values (like with usual approach). To do that we have to redefine a disabled data:

**Definition 7 (Disabled data).** *t is disabled for X in B, if t contains missing values for at least one item i of X, where i does not represent an association with a missing value.*

Example

Data 1: X1=a, X2=? , X3=a.

Data 1 is not disabled for the itemset {X1=a & X2=?} (the missing value for X2 could be informative).

Data 1 is still disabled for {X1=a & X2=c} and {X1=b & X2=c} (See Sect. 4.1).

If a missing value is informative for an attribute, itemsets which test missing values will be frequent. In the other cases, they will disappear because they will decrease under the user specified support (like with usual approach).

In the next section we propose a simulation with real databases where we compare the two approaches.

## 5 Results

### 5.1 Simulation

The simulation consists applying the association rules program, developed from [2] (see Sect. 2), to a database (without missing values) to obtain a ruleset called *original ruleset*. Next 5% of random missing values per attribute are introduced and the program is rerun with [2] and our new approach. The second step is repeated with 10, 15, 20, 25 and 30% of missing values per attribute. We compare results regarding the number of retrieved rules, i.e rules which were present in the original ruleset, and the number of new rules, i.e rules which were not in the original ruleset.

This simulation is made with two databases. The first database has been generated with Synthetic Classification Data Sets (SCDS) program<sup>2</sup>, developed by Gabor Melli. This program generates synthetic data sets which are particularly useful to test Knowledge Discovery from Database (KDD) algorithms. This database has 2000 data. Each data contains 8 attributes with 4 different possible values<sup>3</sup>. The second database is a classical one in machine learning community. Vote is composed by the Voting records drawn from the Congressional Quarterly Almanac, 98th Congress, 2nd session 1984, Volume XL: Congressional Quarterly Inc. Washington, D.C., 1985. This database has 435 data, each with 16 attributes with 3 different possible values.

For the SCDS database we have used a minimum support of 10% and a minimum confidence of 80%. For the vote database we have used a minimum support of 40% and a minimum support of 80%. In our simulation, conclusions of the rules, i.e Y in a rule  $X \rightarrow Y$ , are items (not itemsets) to be more accurate rules.

Results are presented in Fig. 4. The new approach is more robust to the

---

<sup>2</sup> <http://fas.sfu.ca/cs/people/GradStudents/melli/SCDS/>

<sup>3</sup> The command line to generate this database is scds -v -O2000 -R3 -A8 -d4 -C5 -D2

Database SCDS, original ruleset: 2975 rules.

| % of missing values | 5    | 10   | 15   | 20   | 25   | 30   |
|---------------------|------|------|------|------|------|------|
| Usual approach      |      |      |      |      |      |      |
| Retrieved rules     | 394  | 123  | 68   | 33   | 0    | 0    |
| Missing rules       | 2581 | 2852 | 2907 | 2942 | 2975 | 2975 |
| New rules           | 0    | 1    | 0    | 1    | 0    | 0    |
| New approach        |      |      |      |      |      |      |
| Retrieved rules     | 2816 | 2752 | 2603 | 2509 | 1773 | 1784 |
| Missing rules       | 159  | 223  | 372  | 466  | 1202 | 1191 |
| New rules           | 36   | 29   | 57   | 54   | 20   | 0    |

Database Vote, original ruleset: 216 rules.

| % of missing values | 5   | 10  | 15  | 20  | 25  | 30  |
|---------------------|-----|-----|-----|-----|-----|-----|
| Usual approach      |     |     |     |     |     |     |
| Retrieved rules     | 41  | 4   | 1   | 0   | 0   | 0   |
| Missing rules       | 175 | 212 | 215 | 216 | 216 | 216 |
| New rules           | 0   | 0   | 0   | 0   | 0   | 0   |
| New approach        |     |     |     |     |     |     |
| Retrieved rules     | 211 | 145 | 178 | 155 | 137 | 121 |
| Missing rules       | 5   | 71  | 38  | 61  | 79  | 95  |
| New rules           | 48  | 13  | 46  | 100 | 54  | 30  |

Fig. 4. Comparison between usual and new approaches

missing values. An important point is that new rules discovered by the new approach are not false. In fact a vdb for a rule could not be exactly equivalent to the original database. Actually support and confidence are true with an epsilon error. With Vote (resp. SCDS) , new rules are present in a original ruleset with support of 38% (resp. 8%) and confidence of 78% (resp. 78%).

## 5.2 Medical Experimentation

We use association rules to discover useful relations in a database concerning the Hodgkin disease. The Hodgkin disease is a cancer. At the present time, there are several efficacious treatments according to the stage of disease. The consequences are important (leukemia ...) for last stages. The aim of this experiment is to try to define more accurately the different stages of the disease to apply the best treatment.

The database is collected by the lymphome group of the O.E.R.T.C (European Organization of Research and Treatment of Cancer). At the present time, it is managed by the François Baclesse center of Caen and had involved 2460 people since 1964. We have studied the H7 protocol concerning 832 people (1988-1993). There are 27 attributes and 11 of them have missing values (52%, 43%, 88%, 84%, 39%, 8%, 6%, 7%, 2%, 6% and 6.5%).

According to a minimum support of 50% and a minimum confidence of 80% we have found, with the usual treatment, 17133 rules. With our new treatment we have found 51286 rules (three times more). Previous simulations makes one think that new rules are useful. At present we work with experts to evaluate them. First results are of a great interest.

## 6 Conclusion

Association rules are very useful to explore very large transaction databases and are a success. Many of real-world application have their data managed in relational database where the problem of missing attribute values is often inevitable. The missing values problem has never been considered of interest in association rules algorithms because they have been first developed to mine association from transaction databases where practically it does not exist. However it becomes a problem if we try to mine associations in relational databases containing missing values using fast algorithms of associations rules [2] [16] [15] [10]: generally missing values can not be treated like a proper attribute value [9] [12] [6]. In this paper we have proposed a new way to deal efficiently with informative and random missing values, completely compatible with [2]. The results of Sect. 5 show that this approach is much more robust than the older one and outperforms it by factors five to ten. In our real world experiment we have used this approach with a medical databases containing missing values to help experts: with the usual treatment we have found 17133 rules, and with the new one we have found 51286 rules.

In the future, an interesting application could be to explore data containing missing values and to use strong rules, i.e with high confidence, to fill missing values. This method could replace advantageously usual treatments of missing values in decision tree where they are filled with poor confidence and where they are often replaced by noise [14] [13] [6] [5].

## References

1. R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In Proc. of the ACM SIGMOD Conference on Management of Data, Washington, D.C., p 207-216, May 1993.
2. R. Agrawal, H. Mannila, R. Srikant, H. Toivonen and A. I. Verkamo. Fast Discovery of Association Rules. In Advances in Knowledge Discovery and Data Mining, Chapter 12, AAAI/MIT Press, 1996.
3. R. Agrawal, R. Srikant. Fast algorithms for mining association rules in large databases. In Proc. of the Twentieth Int'l Conference on Very Large Databases (VLDB'94), p. 487 - 499, September 1994.
4. K. Ali, S. Manganaris, R. Srikant: Partial Classification using Association Rules, in Proc. of the 3rd Int'l Conference on Knowledge Discovery in Databases and Data Mining, Newport Beach, California, August 1997.
5. L. Breiman, J.H Friedman, R.A Olshen, C.J Stone. Classification and Regression Trees, Wadsworth Int'l Group, Belmont, CA, The Wadsworth Statistics/Probability Series, 1984.

6. G. Celeux. Le traitement des données manquantes dans le logiciel SICLA. Technical reports number 102. INRIA, France, December 1988.
7. B. Crémilleux, C. Robert. A pruning method for decision trees in uncertain domains: applications in medicine. Int'l Workshop on Intelligent Data Analysis in Medicine and Pharmacology, ECAI 1996, p 15-20, Budapest 1996.
8. B. Crémilleux, C. Robert. A theoretical framework for decision trees in uncertain domains: application to medical data sets. 6th Conference in Artificial Intelligence in Medicine Europe (AIME 97), p 145-156, Grenoble 1997.
9. W.Z Liu, A.P White, S.G Thompson and M.A Brammer. Techniques for Dealing with Missing Values in Classification. In Second Int'l Symposium on Intelligent Data Analysis, Birkbeck College, University of London, 4th-5th August 1997.
10. H. Mannila, H. Toivonen and A. Ikeri Verkamo. Efficient algorithms for discovering association rules. In Knowledge Discovery in Databases, Papers from the 1994 AAAI Workshop (KDD'94), p. 181-192, Seattle, Washington, July 1994.
11. J.R Quinlan. Induction of decision trees. Machine learning, 1, p. 81-106, 1986.
12. J.R Quinlan. Unknown Attribute Values in Induction, in Segre A.M.(ed.), Proc. of the Sixth Int'l Workshop on Machine Learning, Morgan Kaufmann, Los Altos, CA, p. 164-168, 1989.
13. J.R Quinlan. C4.5: Programs for Machine Learning, Morgan Kaufmann, San Mateo, CA, 1993.
14. A. Ragel: Traitement des valeurs manquantes dans les arbres de décision. Technical reports, Les cahiers du GREYC. University of Caen, France, 1997.
15. A. Savasere, E. Omiecinski, S. Navathe. An efficient algorithm for mining association rules in large databases. Proc. of the 21st Int. Conference on Very Large Databases (VLDB'95), p. 432 - 444, Zurich, Switzerland, 1995.
16. H. Toivonen. Sampling large databases for association rules. In Proc. of the 22nd Int'l Conference on Very Large Databases (VLDB'96), p. 134-145, Bombay, India, 1996

# Mining Regression Rules and Regression Trees<sup>\*</sup>

Buh-Yun Sher<sup>1</sup>, Shin-Chung Shao<sup>2</sup> and Wen-Shyong Hsieh<sup>3</sup>

<sup>1</sup>Institute of Electrical Engineering, National Sun Yat-Sen University  
Kaohsiung, Taiwan, 804, ROC  
[bysher@cc.cma.edu.tw](mailto:bysher@cc.cma.edu.tw)

<sup>2</sup>Department of Information Management, NanHwa Management College  
Da-Lin, Chia-Yih, Taiwan, 622, ROC  
[scshao@acm.org](mailto:scshao@acm.org)

<sup>3</sup>Institute of Computer and Information Engineering, National Sun Yat-Sen University  
Kaohsiung, Taiwan, 804, ROC  
[wshsieh@mail.nsysu.edu.tw](mailto:wshsieh@mail.nsysu.edu.tw)

**Abstract.** We propose a new type of regression rules to represent the conditional functional relationship between a response variable and  $p$  numeric-valued explanatory variables, conditioning on values of a set of categorical variables. Regression rules are ideal for representing relationships existed in mixture of categorical data and numeric data. A set of regression rules can also be presented in the form of a tree graph, called the regression tree, to assist understanding, interpreting, and applying these rules. We also introduce a process for mining regression rules from data stored in a relational database. This process uses the concept of multivariate and multidimensional OLAP to minimize operations for source data retrieval, and uses homogeneity tests to reduce the size of search space. Thus, it can be used to support mining regression rules in an efficient manner in the context of very large databases.

## 1 Introduction

We propose a new type of regression rules to represent the conditional functional relationship between a response variable and a set of numeric-valued explanatory variables. The functional relationship can be modeled as a regression equation, e.g.,  $Y = X\beta$ , where  $Y$  is a column vector denoting the response variable,  $X$  is a data matrix denoting explanatory variables, and  $\beta$  is a column vector denoting regression coefficients to be estimated. The regression equation is conditioned on a conjunctive predicate defined over a set of categorical variables. That is, a regression rule can be represented as:  $Pred(C) \rightarrow Y = X\beta$ , where  $C$  denotes a set of categorical variables, and  $Pred(C)$  denotes a conjunctive predicate over  $C$ . Potential applications of regression rules include the modeling of consumption function, which relates spending to income, conditioning on census or demographic data, and the modeling of demand

---

\* This work was supported by the National Science Council, Republic of China, under Contract NSC 87-2416-H-145-001.

functions, which relates quantity to price, conditioning on contextual information regarding consumer behavior or seasonal effects. For example,  $((\text{Sex} = \text{female}) \wedge (\text{State} = \text{CA})) \rightarrow (\text{Spending} = 600 + 0.75 * \text{Income})$ , and  $((\text{Sex} = \text{male}) \wedge (\text{State} = \text{WY})) \rightarrow (\text{Spending} = 450 + 0.69 * \text{Income})$  are two conditional consumption functions presented in the forms of regression rules.

Regression rules are useful in representing relationships hidden in mixtures of categorical and numeric data, while other kinds of rules, e.g., classification rules [2], [12] and association rules [3], [4], [10], deal mainly with categorical data. In modeling regression rules, we restrict the head of the rule, i.e., a conjunctive predicate, involves only categorical variables, and explanatory variables must be numeric. Values of the response variable can be either numeric or categorical. Thus, a family of regression rules, including linear regression rules, quadratic regression rules, logistic regression rules, and lognormal regression rules can be derived by using the linear regression models or the generalized linear models [1], [5]. Therefore, the feature of *conditioning on predicate of categorical variables* and *versatile modeling capabilities* makes regression rules an ideal scheme for knowledge representation. Moreover, a set of regression rules and their relationships can be presented in a tree graph, called the regression tree, to assist understanding, interpreting, and applying these rules.

This paper is organized as follows. In Section 2 we will motivate our research by providing an example of mining conditional demand functions for a retail chain store. The constructs of regression rules and regression trees are then introduced. We will also sketch the process for mining regression rules from source data stored in a relational database in Section 2. Section 3 is a review of our previous research results on online analytical processing (OLAP), and its relevance to mining regression rules. We will show that mining regression rules in the context of very large databases can be made more efficient using our multivariate and multidimensional OLAP approach. Section 4 will introduce statistical methods for mining regression rules. In particular, the homogeneity tests will be introduced to illustrate how to reduce the size of search space. As will be seen in Section 2, the number of regression rules grows exponentially with the number of categorical variables. Therefore, reducing search space has important implications to the efficiency of mining regression rules. Section 5 concludes this paper.

## 2 Regression Rules and Regression Tree

In this section, we will first illustrate a motivating example of deriving demand functions for a retail chain store. We will show that the methods commonly used in applied statistics and in data mining are not suitable for this kind of problems. We then introduce the constructs of regression rules and regression tree, and how we use them to represent conditional demand functions, or more general, conditional regression equations. The three-stage process of mining regression rules is also sketched. Detailed techniques and methods used in these stages will be introduced in Section 3 and Section 4.

## 2.1 Motivating Example

For illustrative purpose, we consider the problem of deriving *demand functions* for a retail chain store. Demand functions relate quantities to unit prices, and are useful in determining pricing policy. Suppose the database of a large retail chain store maintains a POS (point-of-sale) table in which seven attributes <Date, Area, StoreID, ItemID, Category, Price, Quantity> are defined. Each row in POS keeps records of an individual sale transaction of an item, belonging to a category, at a specified unit price and quantity, made in a branch store, located in an area, on a given date. The unit price of the same item may vary due to different discount rates, coupons, and promotion plans. Our purpose is to find the functional relationships between quantities and prices for different combinations of areas, categories, and time periods (in terms of month), i.e., demand functions under different conditions.

A naïve approach of deriving such demand functions is to perform a regression analysis of Quantity on Price, e.g., a linear regression with model  $Q = \alpha + \beta * P$ , where  $Q$  and  $P$  denotes Quantity and Price,  $\alpha$  and  $\beta$  denote intercept and slope of the regression line, respectively. However, the above model ignores the effects of categorical variables to demand functions and thus important contextual information are lost. To introduce categorical variables into the above model, a technique called *regression with dummy variables* can be used [11]. That is, we may assign numeric values to distinct values of categorical variables, e.g., Area, Category and Month. Then the model for demand functions can be specified as:

$$\text{Quantity} = \alpha + \beta_1 * \text{Area} + \beta_2 * \text{Category} + \beta_3 * \text{Month} + \beta_4 * \text{Price} \quad (1)$$

One of the major problems of regression with dummy variables is that using the above model, we are implicitly assuming that the *intercept* of the demand functions changes for different combinations of Area, Category, and Month, but the *slope* stays the same [11]. To overcome this problem, a better strategy is to run a separate regression for each distinct combination of levels of categorical variables. However, if the number of categorical variables and/or the number of levels of these categorical variables is/are large, then the number of regression equations to be estimated will be too large to be executed. In general, the number of regression equations grows exponentially with the number of categorical variables (see Section 2.2). Moreover, the resulting regression equations may not be presented in understandable forms.

Another approach dealing with the problem of exploring relationships among mixtures of categorical variables and numeric variables is the tree-based models [2], [5]. Tree-based models find the relationship between a response variable and a set of explanatory variables. Both response variable and explanatory variables can be categorical or numeric. Tree-based models are frequently used in KDD to mine classification rules [2], [6], [7], [12]. Fig. 1(a)<sup>1</sup> illustrates an example of a classification tree,

---

<sup>1</sup> The example of mileage/automobile weights and Fig. 1(a), Fig. 1(b) are adopted from Chapter 9 in [5].

which relates mileage (response variable) to automobile weight. The classification rules can be read from root node of the tree to a leaf node. For example, the rule represented by the rightmost path and by the leftmost path in Fig. 1(a) can be written respectively as: “If weight > 3637.5 then mileage = 18.67”, and “If weight < 2280 then mileage = 34.0”.

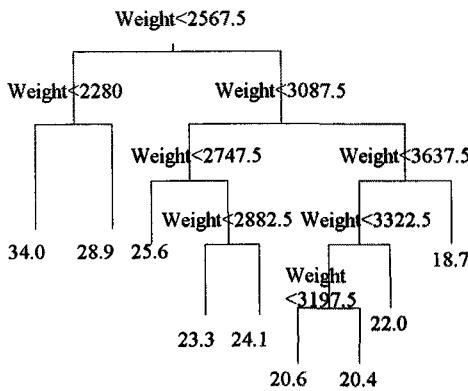


Fig. 1(a). Classification Tree.

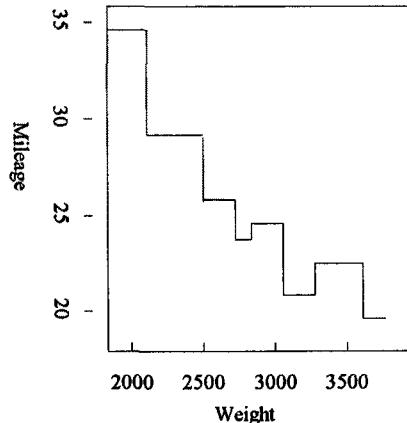


Fig. 1(b). Plot of Step Function.

Tree-based models present relationships between values of the response variable and *discrete splits* of values of explanatory variables so that the fitted model is a *step function*, as illustrated in Fig. 1(b). Therefore, for all weights in the same range, say,  $2280 \leq \text{weight} < 2567.5$ , the predicted values of mileage are the same, i.e., 22.9. Thus, it is not ideal for representing relationships between numeric-valued variables.

## 2.2 Constructs of Regression Rules and Regression Trees

Let  $T$  denote the table containing source data from which regression rules are derived.  $T$  defines three set of variables<sup>2</sup>,  $T = \{C, Y, X\}$ , where  $C$  is a set of categorical variables which take nominal, ordinal or interval values from finite domains,  $Y$  is the response variable, and  $X$  is a set of explanatory variables whose elements take numeric values. Without loss of generality, we assume that  $X$  is an  $n \times p$  matrix containing numeric data, and  $Y$  is an  $n \times 1$  column vector. A regression rule  $\mathfrak{R}$  can be represented as:

$$\mathfrak{R}: \text{Pred}(C) \rightarrow Y = X\beta \quad \text{with confidence } r \quad (2)$$

where  $\text{Pred}(C)$  denotes a conjunctive predicate over  $C$ ,  $Y = X\beta$  is a model of linear regression which relates  $Y$  to  $X$ ,  $\beta$  is a  $p \times 1$  column vector containing regression

<sup>2</sup> We will interchangeably use *variables*, *attributes*, and *columns* to refer to the same thing, columns of a table.

coefficients to be estimated, and  $r$  denotes the *confidence measure* of the rule, which measures the goodness-of-fit of the model. Following are examples of regression rules derived from the POS table:

- (1)  $(\text{Area} = \text{North}) \wedge (\text{Category} = \text{Toys}) \wedge (\text{Month} = \text{Dec}) \rightarrow Q = 24110 - .021 * P$   
with  $r = 0.87$
- (2)  $(\text{Area} = \text{South}) \wedge (\text{Category} = \text{Toys}) \wedge (\text{Month} = \text{Dec}) \rightarrow Q = 19120 - .033 * P$   
with  $r = 0.84$
- (3)  $(\text{Area} = \text{North}) \wedge (\text{Category} = \text{Toys}) \rightarrow Q = 1971 - .56 * P$  with  $r = 0.79$

The above rules say that in northern area the demand of toys in December (Christmas season) is significantly higher than that in the whole year, and the demand of toys in northern area is less sensitive to changes of prices in December than that of southern area in December.

Since there may be many of such rules, we can represent all these rules in the form of a tree graph, called the *regression tree*. A regression tree is a tree with  $c_i \in C$  as non-leaf nodes, levels of  $c_i$  as edges, and regression equations as leaf nodes. To deal with cases of missing values and skipping categorical variables in predicates, we include an additional level ALL to the domain of each categorical variable to denote *all* or *any* value of that variable. Thus, each non-leaf node  $c_i$  has  $(l_i + 1)$  edges originating from it, where  $l_i$  denotes the number of levels of  $c_i$ . Fig. 2 illustrates the regression tree derived from our example.

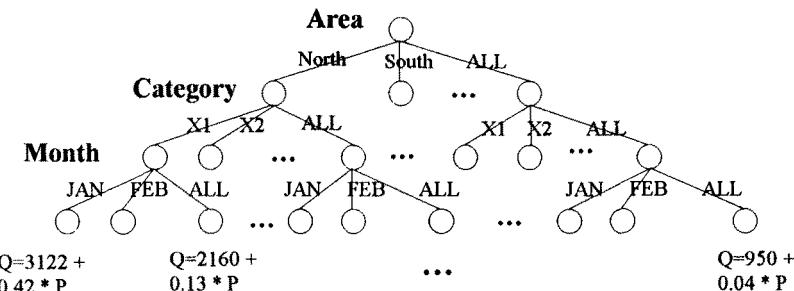


Fig. 2. An Example of the Regression Tree Derived From POS

A regression rule can be read along the path from the root node to a leaf node. In Fig. 2, the two leaf nodes on the left-hand-side with specified regression equations can be read as: “If ( $\text{Area} = \text{North}$ ) and ( $\text{Category} = \text{X}1$ ) and ( $\text{Month} = \text{Jan}$ ) then  $Q = 3122 + 0.42 * P$ ”, and “If ( $\text{Area} = \text{North}$ ) and ( $\text{Category} = \text{X}1$ ) then  $Q = 2160 + 0.13 * P$ ”. In the later case, the categorical variable Month does not appear in the predicate of the rule. Thus, the later rule can be used to predict  $Q$  when value of Month is missing. The rightmost leaf node in Fig. 2 is equivalent to the case of directly performing a regression of Quantity on Price, without taking contextual information into consideration.

A serious problem with mining regression rules is that the number of regression rules, i.e., the total number of leaf nodes in a regression tree, is given by

$$N = \prod_{i=1}^m l_i + 1 \quad (3)$$

where  $m$  is the number of categorical variables and  $l_i$  denotes the number of levels of the  $i$ -th categorical variable. For example, suppose the store has branch stores located in 10 different areas and maintains stocks belonging to 1000 categories. We want to take Month as one of the categorical variables, then the number of leaf nodes in the regression tree with  $C = \{\text{Area, Category, Month}\}$  is given by  $11 \times 1001 \times 13 = 143,143$ . In Section 4 we will introduce how we use homogeneity tests to reduce the size of search space and to increase the performance of mining regression rules.

### 2.3 Mining Regression Rules

The command syntax for mining regression rules from data stored in a relational database is:

```
SELECT Attribute list
FROM table(s)
[WHERE conditions]
MINE REGRESSION RULE Y ~ X
WITH CONFIDENCE t
GROUP BY C
```

where *Attribute list* in the SELECT clause specifies C, Y and X. The FROM and the WHERE clauses specify the table name(s) and search conditions. The GROUP BY clause is used to partitioning levels of categorical variables. Notice that the SELECT clause, the FROM clause, the WHERE clause, and the GROUP BY clause are similar to those in SQL syntax. Indeed, we will transform these clauses into corresponding SQL commands to derive aggregated data from OLAP as inputs to mining process (See Section 3). The  $Y \sim X$  appeared in the fourth line of the above command syntax can be read as Y is modeled as a linear function of X. The confidence measure  $t$  is used as a threshold to filter out insignificant rules. A regression rule with confidence measure  $r$  is said as *significant* if  $r \geq t$ , otherwise, it is said as *insignificant*. The following command mines regression rules in our example:

```
SELECT Area, Category, Month = month(Date), Quantity, Price
FROM POS
MINE REGRESSION RULE Quantity ~ Price
WITH CONFIDENCE 0.8
GROUP BY Area, Category, Month
```

where  $\text{month}( )$  denotes the time function used to derive the month of a date data type. The process for mining regression rules can be divided into three stages: (1) the (source) data preprocessing stage, (2) the homogeneity tests stage, and (3) the estimation stage. In the first stage, a set of multivariate and multidimensional aggregated

data are derived in the form of multivariate and multidimensional OLAP. This set of data provides sufficient information to mine regression rules so that there is no need to retrieve source data in other stages. This *one-pass scanning of source data* feature has important implications to the efficiency of data mining in the context of very large databases. Data preprocessing stage will be illustrated in greater details in Section 3. In the second stage, a likelihood ratio test and a multivariate analysis of variance (MANOVA) are performed to test if there is any categorical variable *independent* to Y and X (to be defined in Section 4). If such independent categorical variables are found, then the size of search space can be significantly reduced. In the third stage, a regression rule is derived for each leaf node on the regression tree. The homogeneity tests stage and the estimation stage are subjects of Section 4.

### 3 Multivariate and Multidimensional OLAP

The purpose of this section is to illustrate how to preprocess source data with the aim at one-pass scanning of source data. *One-pass scanning of source data* refers to scanning the file containing source data at most once in the entire mining process. In our previous research [13], we proposed a multivariate and multidimensional OLAP model, which can be used to support mining regression rules and regression trees in a one-pass scanning manner. The creation, materialization and manipulation of *multivariate aggregate view* (MAV) will be illustrated in this section.

#### 3.1 Definition and Creation of Multivariate Aggregate View

An MAV is a materialized view containing multivariate and multidimensional aggregated data. An MAV is defined by  $\langle T, C, Z \rangle$ , where T is a table, C is a set of  $m$  dimension attributes, and Z is a set of  $(p + 1)$  measure attributes. An MAV contains a table defining  $m + 1 + (p + 1) + ((p+ 1)^2 + (p + 1))/2$  attributes. The first  $m$  attributes are for dimension variables; the next attribute is for Count; the following  $(p + 1)$  attributes are for Sum, one for each measure attribute; the rest attributes are for  $Z'Z$ . Since  $Z'Z$  is symmetric, we represent it using  $((p+ 1)^2 + (p + 1))/2$  attributes. Notice that  $Z'Z$  can be partitioned into  $YY$ ,  $YX$ ,  $X'Y$  and  $X'X$ , which will be used in perform homogeneity tests and estimation of regression rules.

The creation, derivation and materialization of an MAV can be executed by an SQL command. For example, the following command creates an MAV, which will be used to support mining regression rules and regression tree in our example:

```

SELECT Area, Month = month(Date), Category, _COUNT = COUNT(*),
 SUM_Q = SUM(Quantity), SUM_P = SUM(Price),
 SS_Q = SUM(Quantity ^ 2), SS_P = SUM(Price ^2),
 SCP_Q_P = SUM(Quantity * Price)
INTO MAV_POS
FROM POS
GROUP BY Area, Month, Category

```

In the above command, SS\_Q and SS\_P denote sums of squares of quantities and prices, which are diagonal elements of  $Z'Z$ , SCP\_Q\_P denotes sum of cross-products between quantities and prices, which are off-diagonal elements of  $Z'Z$ .

It is noted that the SELECT ... FROM ... [WHERE ...] GROUP BY ... clauses in the command syntax for mining regression rules (see Section 2.3) are very similar to the command syntax for creating MAV. The transformation of a command for mining regression rules to a MAV creation command follows in a straightforward manner. Thus, in the data preprocessing stage, the command for mining regression rules is transformed into the command for creating an MAV.

### 3.2 Manipulation of Multivariate Aggregate View

Since MAV is a derived table (materialized aggregate view) stored in a relational database, its records can be manipulated using SQL commands. Each record in MAV corresponds to a leaf node in a regression tree whose path does not involve edges labeled by ALL. For example, the record in MAV with Area = “North”, Category = “X1” and Month = “Jan” corresponds to the leftmost leaf node in Fig. 2. Retrieving individual records, one at a time, from MAV to estimate regression rules is an obvious task, e.g., using cursor to browse each record in an MAV.

In performing homogeneity tests and in estimating regression rules whose paths involve one or more edges labeled by ALL, it is required to further aggregating values in different rows. As an extreme case, consider the rightmost rule in Fig. 2. Estimating  $\beta$  for that rule requires count ( $n$ ), summations and the  $Z'Z$  derived from the entire POS table. Due to the *distributive property* of aggregate functions [], aggregated data in different records can be added up. An aggregate function  $f()$  is said to be distributive if  $f(X) = f(X_1) + f(X_2) + \dots + f(X_k)$ , where  $X_1, X_2, \dots, X_k$  vertically partition  $X$ . In particular, we consider Count and Sum. It is easy to see that  $\text{Count}(X) = \text{Count}(X_1) + \text{Count}(X_2) + \dots + \text{Count}(X_k)$  and  $\text{Sum}(X) = \text{Sum}(X_1) + \text{Sum}(X_2) + \dots + \text{Sum}(X_k)$ . Since Count and Sum are distributive and all aggregated data maintained in MAV are derived from Count and Sum, it follows that aggregated data in different rows of the same MAV can be added up to satisfy the need to estimating regression equations for rules involving edges labeled by ALL. For example, the  $Z'Z$  derived from all records in POS satisfying Month = “January” and Area = “North” can be derived from the following command:

```
SELECT SUM(SS_P), SUM(SS_Q), SUM(SCP_P_Q)
FROM MAV_POS
WHERE Area = "North" and Month = "Jan"
```

The results of the above query can be used to mine regression rule having the predicate  $((\text{Area} = \text{“North”}) \wedge (\text{Month} = \text{“Jan”}))$ . This rule corresponds to the path following Area = “North”, Category = ALL, and Month = “Jan”.

## 4. Derivation of Regression Rules and Regression Trees

In this section we will show how the second stage and the third stage in the mining process, namely homogeneity tests and estimation of regression coefficients, is performed. In Section 4.1 we will introduce how we use the generalized likelihood ratio tests and multivariate analysis of variance (MANOVA) to test whether we can reduce the search space. In Section 4.2 we will present the unbiased least squares estimation of regression coefficients. Due to space limitation, we only present specific statistical methods we will use in those stages, and the rationale of using them. Detailed descriptions about these statistical methods can be found in cited references. Readers can find that all these statistical methods require only data contained in MAV.

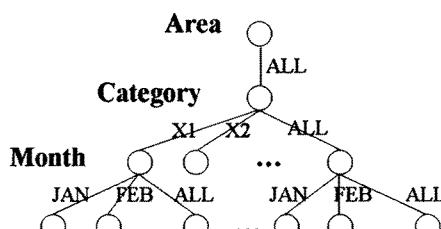
### 4.1 Reducing Search Space

Let  $c_i \in C$ , and  $c_i$  has  $k$  levels that partition  $Z$  into  $k$  submatrices denoted by  $Z_1, Z_2, \dots, Z_k$ .  $c_i$  is said to be *independent* to  $Z$  if  $Z_1, Z_2, \dots, Z_k$  are generated from the same population. Assuming that  $Z_j$  has a multivariate normal distribution with mean vector  $\mu_j$  and covariance matrix  $\Sigma_j$ , i.e.  $Z_j \sim N(\mu_j, \Sigma_j)$ ,  $j = 1, 2, \dots, k$ . Since normal distribution defines the first two moments, it follows that  $c_i$  is independent to  $Z$  if the following null hypotheses are true:

$$^1H_0 : \Sigma_1 = \Sigma_2 = \dots = \Sigma_k \quad (4)$$

$$^2H_0 : \mu_1 = \mu_2 = \dots = \mu_k \quad (5)$$

If  $c_i$  is independent to  $Z$  then we may remove  $c_i$  from  $C$ , which reduces the search space by one dimension and a significant performance gains can be achieved. For instance, if we found that Area is independent to {Quantity, Price}, then the number of regression rules to be estimated is reduced from 143,143 to 13,013 (See Section 2.2). The regression tree in Fig. 2 is reduced to the tree illustrated in Fig. 3 (regression equations are not shown). Therefore, in the second stage of mining regression rules the two null hypotheses ' $H_0$ ' and ' $H_0$ ' are tested against each categorical variable. All independent categorical variables are removed from  $C$  before proceeding to the third stage. The test of ' $H_0$ ' can be done by performing a generalized likelihood-ratio test, and the test of ' $H_0$ ' can be done by performing an MANOVA, which are sketched as follows [9].



**Fig. 3. A Reduced Regression Tree**

### *Generalized Likelihood-Ratio Test for Equality of Covariance Matrices*

We assume that  $Z_I, I = 1, 2, \dots, k$ , be an  $(n_I \times p)$  data matrix which is generated from  $p$ -dimensional multivariate normal populations. Let  $\Sigma_I$  denote the covariance matrix of  $Z_I$ ,  $S_I$  be the unbiased estimate of  $\Sigma_I$ , and

$$S = \frac{1}{n} \sum_{I=1}^k n_I S_I \quad (6)$$

is the pooled estimate of the common covariance matrix. The test statistic  $MC'$  is approximately distributed as a Chi-squared distribution with degrees of freedom  $\frac{1}{2}(k-1)p(p+1)$ , as  $n_I$  becomes large, we have:

$$M = \sum_{I=1}^k n_I \ln|S| - \sum_{I=1}^k n_I \ln|S_I| \quad (7)$$

$$C^{-1} = 1 - \frac{2p^2 + 3p - 1}{6(p+1)(k-1)} \left( \sum_{I=1}^k \frac{1}{n_I} - \frac{1}{n} \right) \quad (8)$$

where  $|S_I|$  denotes the determinant of the covariance matrix  $S_I$ . Thus, we can calculate  $MC'$  and test ' $H_0$ ' using Chi-squares distribution. [9] introduces another method for testing ' $H_0$ ' based on  $F$  approximation, c.f. [9] for comparing Chi-square test and  $F$ -test.

### *MANOVA: Testing Equality of Mean Vectors*

Suppose ' $H_0$ ' is accepted, that is,  $Z_1, Z_2, \dots, Z_k$  have a common covariance matrix  $\Sigma$ , then we may use MANOVA to test ' $H_0$ '. In large sample if ' $H_0$ ' is true, the test statistic

$$F = \frac{(1-y) m_2}{y m_1} \quad (9)$$

has an  $F$  distribution with  $m_1$  and  $m_2$  degrees of freedom, where

$$y = \Lambda^{1/2}, \quad s = \sqrt{\frac{p^2(k-1)^2 - 4}{p^2 + (k-1)^2 - 5}}, \quad m_1 = p(k-1) \quad (10)$$

$$m_2 = s[n - (p+k+2)/2] - \frac{p(k-1)}{2} + 1, \quad \Lambda = |\mathbf{W}| / |\mathbf{W}+\mathbf{G}| \quad (11)$$

where  $\mathbf{W}$  and  $\mathbf{G}$  denote *within group sum of squares matrix* and *among group sum of squares matrix*, respectively, and  $|\cdot|$  denotes determinant of a matrix. The above

approximation is usually referred to as *Rao's F* [9].

Supporting the computation in the second stage requires  $n$ ,  $n_i$ (row count of each submatrix), mean vectors  $\mu_i$ , and sample covariance matrix  $S$ . It is obvious that all the above statistics can be derived from an MAV derived in the first stage.

## 4.2 Unbiased Least-Square Estimation of Regression Equations

In this subsection, we present important results of multiple regression models. Consider the linear model:

$$Y = X\beta + \varepsilon \quad (12)$$

where  $\varepsilon$  is an  $(n \times 1)$  column vector denoting error terms or residuals. The unbiased least-squares estimation of the regression coefficient  $\beta$  is given by:

$$\hat{\beta} = (X'X)^{-1}(X'Y) \quad (13)$$

To measure the goodness-of-fit of the model, adjusted R-squares is commonly used, which can be calculated by:

$$\bar{R}^2 = 1 - \left( \frac{\hat{\varepsilon}'\hat{\varepsilon}}{Y'Y - n\bar{Y}^2} \right) \left( \frac{n-1}{n-p} \right) \quad (14)$$

where  $\hat{\varepsilon}'\hat{\varepsilon} = Y'Y - \hat{\beta}'X'Y$ . We will use adjusted R-squared as confidence measure  $r$  for regression rules. Since  $X'X$ ,  $X'Y$  and  $Y'Y$  are submatrices of  $Z'Z$ , which are maintained in an MAV derived in the first stage, it follows that mining regression rules can be fully supported by MAV. It is noted that other test statistics regarding the distribution of regression coefficients, e.g.,  $t$ -test,  $\chi^2$  test,  $F$ -test, can also be supported by MAV. Due to space limitation, we skip detailed discussion of derivation of these test statistics.

## 5. Conclusion

In this paper we proposed a new types of regression rules to represent conditional functional relationships between a response variable  $Y$  and  $p$  numeric-valued explanatory variables  $X$ , and a process for mining regression rules from source data stored in a relational database. Regression rules can be presented in a tree graph for better understanding and interpretation.

The regression rules we have shown in this paper, called linear regression rules, is a special case of a family of regression rules. Other types of regression rules can also be estimated using, say, generalized linear models, and presented in the form of regression tree, which are the subjects of our future research. Therefore, regression

rule is an ideal scheme for representing conditional functional relationships among mixture of categorical data and numeric data.

## References

1. A. Agresti: *Categorical Data Analysis*, John Wiley & Sons, 1990.
2. L. Breiman, J. Friedman, R. Olshen and C. Stone: *Classification and Regression Trees*, Wadsworth & Brooks, Pacific Grove, CA, 1984.
- 3.S. Brin, R. Motwani and C. Silverstein: *Beyond Market Baskets: Generalizing Association Rules to Corrections*, in SIGMOD RECORD Vol. 26, No. 2, June 1997.
- 4.S. Brin, R. Motwani, J. D. Ullman and S. Tsur: *Dynamic Itemset Counting and Implication Rules for Market Basket Data*, in SIGMOD RECORD Vol. 26, No. 2, June 1997.
- 5.J. Chambers and T. J. Hastie (editors): *Statistical Models in S*, Wadsworth & Brooks/Cole Computer Science Series, 1992.
- 6.Ming-Syan Chen, Jiawei Han and Philip S. Yu: *Data Mining: An Overview from a Database Perspective*, IEEE Transactions on Knowledge and Data Engineering, Vol. 8, No. 6, December 1996, p.p. 866-883.
- 7.U. Fayyad, G. P. Shapiro, P. Smyth and R. Uthurusamy (editors): *Advances in Knowledge Discovery and Data Mining*, AAAI Press/The MIT Press, 1996.
- 8.J. Gray, A. Bosworth, A. Layman and H. Pirahesh: *Data Cube: A relational aggregation operator generalizing group-By, cross-tabs and subtotals*, In Proc. of the 12<sup>th</sup> Int'l Conference on Data Engineering, pp. 152-159, 1996.
- 9.J. D. Jobson: *Applied Multivariate Data Analysis, Vol. II: Categorical and Multivariate-Methods*, Springer Texts in Statistics, Springer-Verlag, 1992.
- 10.Rosa Meo, Giuseppe Psaila and Stefano Ceri: *A New SQL-like Operator for Mining Association Rules*, in Proc. Of the 22<sup>nd</sup> VLDB Conference, Mumbai, India, 1996.
11. R. S. Pindyck and D. L. Rubinfeld: *Econometric Models & Economic Forecasts*, Third Edition, McGraw-Hill, 1991.
12. J. R. Quinlan: *Induction of Decision Trees*, Machine Learning, Vol. 1, pp. 81-106, 1986.
13. S. C. Shao: *Multivariate and Multidimensional OLAP*, to appear in Proceedings of the Sixth International Conference on Extending Database Technology (EDBT98), Lecture Notes in Computer Science, Spain, March 1998.

# Mining Algorithms for Sequential Patterns in Parallel : Hash Based Approach

Takahiko Shintani and Masaru Kitsuregawa

Institute of Industrial Science, The University of Tokyo  
7-22-1 Roppongi, Minato-ku, Tokyo, 106 Japan

**Abstract.** In this paper, we study the problem of mining sequential patterns in a large database of customer transactions. Since finding sequential patterns has to handle a large amount of customer transaction data and requires multiple passes over the database, it is expected that parallel algorithms help to improve the performance significantly.

We consider the parallel algorithms for mining sequential patterns on a shared-nothing environment. Three parallel algorithms (Non Partitioned Sequential Pattern Mining(NPSPM), Simply Partitioned Sequential Pattern Mining(SPSPM) and Hash Partitioned Sequential Pattern Mining(HPSPM)) are proposed. In NPSPM, the candidate sequences are just copied among all the nodes, which can lead to memory overflow for large databases. The remaining two algorithms partition the candidate sequences over the nodes, which can efficiently exploit the total system's memory as the number of nodes increased. If it is partitioned simply, customer transaction data has to be broadcasted to all nodes. HPSPM partitions the candidate sequences among the nodes using hash function, which eliminates the customer transaction data broadcasting and reduces the comparison workload. We describe the implementation of these algorithms on a shared-nothing parallel computer IBM SP2 and its performance evaluation results. Among three algorithms HPSPM attains best performance.

## 1 Introduction

Data Mining, also known as knowledge discovery in databases, has attracted strong attention. Because of the progress of data collection tools, large amount of transaction data have been generated, but such data being archived and not used efficiently. Data mining is the method of efficient discovery of useful information such as rules and previously unknown patterns existing between data items embedded in large databases, which allows more effective utilization of existing data.

The problem of mining sequential patterns in a large database of customer transactions was introduced in [1]. A transaction data typically consists of a customer identifier, a transaction identifier, a transaction time associated with each transaction and the bought items per-transaction. By analyzing these customer-transaction data, we can extract the sequential patterns such as “5% of customer who buy both A and B buy C in the next transaction”.

Several algorithms have been proposed to find sequential patterns[1–3]. An algorithm for finding all sequential patterns, named AprioriAll, was presented in [1]. First, AprioriAll discovers all the sets of items (itemset) with a user-specified minimum support (large itemset), where the support is the percentage of customer transactions that contain the itemsets. Secondly, the database is transformed by replacing the itemsets in each transaction with the set of all large itemsets. Lastly, it finds the sequential patterns. It is costly to transform the database. In [3], a graph-based algorithm, DSG(Direct Sequential pattern Generation), was presented. DSG constructs an association graph to indicate the associations between items by scanning the database once, and generates the sequential pattern by traversing the graph. Though the disk I/O cost of DSG is very low, the related information may not fit in the memory when the size of the database is large. In [2], GSP(Generalized Sequential Pattern) algorithm that discovers generalized sequential patterns was proposed. GSP finds all the frequent sequences without transforming the database. Besides, some generalized definitions of sequential patterns are introduced in [2]. First, time constructions are introduced. Users often want to specify maximum and/or minimum time period between adjacent elements. Second, flexible definition of a customer transaction is introduced. It allows a user-specified window-size within which the items can be present. Third, given a user-defined taxonomy (*is-a* hierarchy) over the data items, the generalized sequential pattern, which includes items span different levels of the taxonomy, is introduced. All the previous algorithms for finding sequential patterns are serial algorithms. Finding sequential patterns has to handle a large amount of customer transaction data and requires multiple passes over the database, which requires long computation time. Thus, its computational requirements are too large for a single processor to have a reasonable response time.

In our previous study, we proposed parallel algorithm for mining association rules on a shared-nothing environment, named HPA(Hash Partitioned Apriori)[4]. Association rules are the rules about what items are bought together within a transaction. HPA not only partitions the transaction database but the candidate itemsets among the nodes. Several other parallel algorithms for mining association rules are introduced[5–8]. However, most of these algorithms do not partition the candidate itemsets. Among them, Data Distribution[7], IDD(Intelligent Data Distribution) and HD(Hybrid Distribution)[9] partition the candidate sequences. Since these algorithms exchange all the transaction data among the nodes, a large amount of communications are required. On the other hand, HPA partitions the candidate itemsets using hash function to reduce the amount of communications.

In this paper, we consider the parallel algorithms for mining sequential patterns on a shared-nothing environment. Three parallel algorithms (Non Partitioned Sequential Pattern Mining, Simply Partitioned Sequential Pattern Mining and Hash Partitioned Sequential Pattern Mining) are proposed. In NPSPM, the candidate sequences are just copied among all the nodes. The remaining two algorithms partition the candidate sequences over the nodes, which can efficiently

exploit the total system's memory as the number of nodes increased. If the candidate sequences are partitioned simply, customer transaction data has to be broadcasted to all nodes. HPSPM partitions the candidate itemsets among the nodes using hash function, which eliminates the customer transaction data broadcasting and reduces the comparison workload. We describe the implementation of these algorithms on a shared-nothing parallel computer IBM SP2 and its performance evaluation results. Among three algorithms, HPSPM attains best performance.

The rest of this paper is organized as follows: Section 2 describes the problem of mining sequential patterns. In section 3, we propose three parallel algorithms. Performance evaluations are given in section 4. Section 5 concludes the paper.

## 2 Sequential Patterns

Here, we introduce some basic concepts of sequential patterns, using the formalism in [2]. Let  $\mathcal{I} = \{i_1, i_2, \dots, i_m\}$  be a set of literals, called items. An itemset  $X$  is defined a set of items such that  $X \subseteq \mathcal{I}$ . Let  $\mathcal{D} = \{s_1, s_2, \dots, s_n\}$  be a set of customer-sequences, where each sequence  $s$  is a set of transactions ordered by increasing transaction time corresponds to a sequence,  $s = \{t_1, t_2, \dots, t_k\}$ . We define  $t_j$  an element of the sequence. A transaction  $t$  has a set of items ( $t \subseteq \mathcal{I}$ ), customer identifier, transaction identifier and transaction time. We say a sequence  $s_a = \langle a_1, a_2, \dots, a_x \rangle$  contains a sequence  $s_b = \langle b_1, b_2, \dots, b_y \rangle$  if there exist integers  $i_1 < i_2 < \dots < i_y$  such that  $a_{i_1} \supseteq b_1, a_{i_2} \supseteq b_2, \dots, a_{i_y} \supseteq b_y$ . The sequence  $s$  has support  $\alpha$  in the customer-sequence database  $\mathcal{D}$ , if  $\alpha\%$  of the customer-sequences in  $\mathcal{D}$  contain  $s$ . The problem of mining sequential patterns is to find all sequences that satisfy a user-specified minimum support on the assumption that we are given a set of customer-sequences. These sequences are called frequent sequences.

Here, we explain the GSP algorithm for finding all frequent sequences, proposed in [2]. Since we consider finding the naive sequential patterns, several generalized definitions introduced in [2] are ignored. In the first iteration (pass 1), support-count for each item is counted by scanning the customer-sequence database. Hereafter we prepare a field named support-count for each sequences, which is used to measure how many times the sequences contained in customer-sequences. Since sequences here consist of just single item, each item has a support-count fields. All the items which satisfy the minimum support threshold are picked out. These items are called frequent 1-sequences ( $L_1$ ). Here,  $k$ -sequence is defined a sequence with  $k$  items. In the second iteration (pass 2), the 2-sequences are generated using  $L_1$ . These sequences are called the candidate 2-sequences ( $C_2$ ). Then the support-count of  $C_2$  is counted by scanning the customer-sequence database. Here support-count of the sequence means the number of customer-sequences which contain it. At the end of scanning the customer-sequence database, the frequent 2-sequences ( $L_2$ ) which satisfy minimum support are determined. The following denotes the  $k$ -th iteration, pass  $k$  ( $k \geq 2$ ).

1. Generate the candidate sequence:

The candidate  $k$ -sequences ( $C_k$ ) are generated using large  $(k - 1)$ -sequences ( $L_{k-1}$ ) which were determined in the previous pass (pass  $k - 1$ ).

2. Count support:

The support-count of  $C_k$  are counted by scanning the customer-sequence database  $\mathcal{D}$ .

3. Determine frequent sequence:

The candidate  $k$ -sequences in  $C_k$  are checked for whether they satisfy the minimum support condition or not, the frequent  $k$ -sequences ( $L_k$ ) which satisfy the minimum support are determined.

This procedure terminates when the new frequent sequence becomes empty.

The procedure for generating candidate  $k$ -sequences is as follows: Candidate generation occurs in two steps. First, in the join step, the candidate  $k$ -sequence  $C_k$  are generated by joining  $L_{k-1}$  with  $L_{k-1}$ . If there are the sequences that the subsequence obtained by dropping the first item of  $s_1$  is identical with the subsequence obtained by dropping the last item of  $s_2$ , the candidate sequence is generated by adding  $s_1$  to the last item in  $s_2$ . The added item becomes a separate element if it was a separate element in  $s_2$ , and a part of the last element of  $s_1$  otherwise. Next, in the prune step, delete all of the sequences in  $C_k$  where some of the  $(k - 1)$ -subset of the candidate  $k$ -sequences are not in  $L_{k-1}$ .

### 3 Parallel Algorithms

In this section, we describe the parallel algorithms for finding all the frequent sequences for shared-nothing parallel machines. In the serial algorithm, the count support processing requires the longest computation time, where the customer-sequence database is scanned and a large number of candidate sequences are examined. We designed a parallel algorithms for the count support processing.

If each node can hold all of the candidate itemsets, parallelization is straightforward. By partitioning the customer-sequence database over all the nodes, disk I/O operation can be done in parallel. In NPSPM, the candidate sequences are just copied among all the nodes. In the case where all of the candidate sequences do not fit within the local memory of single node, the candidate sequences are partitioned into the fragments, each of which fits in the memory size of single node. At this time, NPSPM makes multiple passes over the customer-sequence database in one pass. In order to exploit the total system's memory effectively, the remaining two algorithms partition the candidate sequences over the memory space of all the nodes, and count the support-count by exchanging the customer-sequence among the nodes. For simplicity, we assume that the size of the candidate sequences in one pass is larger than the size of local memory of single node but is smaller than the sum of the memory of all the nodes. It is easy to expand this algorithm to handle the candidate itemsets whose size exceeds the sum of the memory of all the nodes.

### 3.1 Non Partitioned Sequential Pattern Mining : NPSPM

In NPSPM, the candidate sequences are just copied among all the nodes, each node determine the frequent sequences by exchanging the support count values among all the nodes. Though each node can work independently in count support processing, each node has to examine all the candidate sequences. Figure 1 gives the behavior of pass  $k$  of the  $p$ -th node, using the notation in Table 1.

**Table 1.** Notation for NPSPM algorithm

|                   |                                                                                                       |
|-------------------|-------------------------------------------------------------------------------------------------------|
| $\mathcal{L}_k$   | Sets of all the frequent $k$ -sequences                                                               |
| $\mathcal{C}_k$   | Sets of all the candidate $k$ -sequences                                                              |
| $ \mathcal{C}_k $ | The size of $\mathcal{C}_k$ in bytes                                                                  |
| $M$               | The size of main memory in bytes                                                                      |
| $\mathcal{D}^p$   | Customer-sequences stored in the local disk of the $p$ -th node                                       |
| $C_k^d$           | Sets of fragments of the candidate $k$ -sequences. Each fragments fits in the local memory of a node. |
| $ C_k^d $         | The size of $C_k^d$ in bytes                                                                          |
| $L_k^d$           | Sets frequent $k$ -sequences derived from $C_k^d$                                                     |

$k \geq 2$

```

 $\mathcal{C}_k :=$ The candidates of size k generated from \mathcal{L}_{k-1}
 $\{C_k^d\} :=$ Partition \mathcal{C}_k into fragments each of which fits in a node's local memory
 $(d=1, \dots, \lceil |\mathcal{C}_k|/M \rceil)$
for $(d=1; d \leq \lceil |\mathcal{C}_k|/M \rceil; d++)$ do
 forall $s \in \mathcal{D}^p$ do
 Increment the support_count of all candidates in C_k^d that are contained in s
 end
 Send the support_count of C_k^d to the coordinator node
 /* Coordinator node determine L_k^d which satisfy user-specified minimum sup- */
 /* port in C_k^d and broadcast L_k^d to all the nodes */
 Receive L_k^d from the coordinator node
end
 $\mathcal{L}_k := \bigcup_d L_k^d$

```

**Fig. 1.** NPSPM algorithm

Each node works as follows:

1. Generate the candidate sequences:
  - Each node generates the candidate  $k$ -sequences ( $\mathcal{C}_k$ ) using the large ( $k - 1$ )-sequences ( $\mathcal{L}_{k-1}$ ).
  - Insert  $\mathcal{C}_k$  into the candidate hash table.
2. Scan the customer-sequence database and count the support-count value:
  - Each node reads the customer-sequence database from its local disk.
  - Increment the support-count of all the candidates in  $\mathcal{C}_k$  that are contained in  $s$ .
3. Determine the large sequences:
  - After reading all the customer-sequences, all node's support-count are gathered into the coordinator node and examined to determine whether the minimum support condition is satisfied or not.

4. If  $L_k$  is empty, the algorithm terminates. Otherwise the coordinator node broadcasts  $L_k$  to all the nodes,  $k := k + 1$ " and goto "1".

If the size of all the candidate sequences exceeds the local memory size of a single node, the candidate sequences are divided into fragments, each of which can fit within the local memory of a single node. Then, the above process is repeated for each fragment. Figure 1, at the beginning of the for-loop, shows the method by which each of the candidate sequences are divided into fragments with each fragment being processed in order.

### 3.2 Simply Partitioned Sequential Pattern Mining : SPSPM

In NPSPM, the candidate sequences are not partitioned but just copied among all the nodes. However the candidate sequences often becomes too large to fit within the local memory of a single node. SPSPM arbitrarily partitions the candidate sequences equally over the memory space of all the nodes. Thus SPSPM can exploit the aggregate memory of the system. SPSPM assigns the candidate sequences equally to all of the nodes in a round robin manner. By round-robin, the candidates are assigned to the nodes in a cyclical manner with the  $i$ -th candidate assigned to node  $i \bmod n$ , where  $n$  is the number of nodes in the sysytem. Since SPSPM partitions the candidate sequences among the nodes, each node has to broadcast the customer-sequences stored in its local disk to all the other nodes for count support processing, while NPSPM requires no such broadcast. Figure 2 shows the behaviour of pass  $k$  by the  $p$ -th node in SPSPM, using the notation in Table 2. Here we assume the size of candidate itemset is smaller than the size of sum of all the processor's memory. Extension of the algorithm to handle much larger candidate itemset is easy. We can divide the candidate sequences into fragments like in NPSPM.

**Table 2.** Notation for SPSPM algorithm

|                 |                                                                 |
|-----------------|-----------------------------------------------------------------|
| $\mathcal{L}_k$ | Sets of all the large $k$ -sequences.                           |
| $\mathcal{C}_k$ | Sets of all the candidate $k$ -sequences.                       |
| $\mathcal{D}^p$ | Customer-sequences stored in the local disk of the $p$ -th node |
| $C_k^p$         | Sets of the candidate $k$ -sequences assigned the $p$ -th node  |
| $L_k^p$         | Sets of large $k$ -sequences derived from $C_k^p$               |

Each node works as follows:

1. Generate the candidate sequences:
  - Each node generates  $\mathcal{C}_k$  using  $\mathcal{L}_{k-1}$  and inserts a part of the candidate itemsets into its own hash table ( $C_k^p$ ). The candidate  $k$ -sequences are assigned to nodes in a round-robin manner.
2. Scan the customer-sequence database and count the support-count value:
  - Each node reads the customer-sequence database  $\mathcal{D}^p$  from its local disk and broadcasts them to all the nodes. For each customer-sequence entry, the support-count is incremented in the same way as in NPSPM, when read from its own disk or received from another nodes.

```

 $k \geq 2$
 $\mathcal{C}_k :=$ The candidates of size k generated from \mathcal{L}_{k-1}
 $\{C_k^p\} :=$ All the candidate k -itemsets, assigned to the p -th node
forall $s \in \mathcal{D}^p$ do
 Broadcast s to all the nodes
 Increment the support-count value of C_k^p contained in s
 Receive the customer-sequence s' sent from the other nodes and increment the
 support-count of all candidates that are contained in s'
end
 $\{L_k^p\} :=$ All the candidates in C_k^p with minimum support
Send L_k^p to the coordinator node
/* Coordinator node make up $\mathcal{L}_k := \bigcup_p L_k^p$ and broadcast to all the nodes */
Receive \mathcal{L}_k from the coordinator node

```

**Fig. 2.** SPSPM algorithm

## 3. Determine the large sequences:

- After reading all the customer-sequences, each node can determine individually whether each candidate in  $C_k^p$  satisfy minimum support condition or not.
- Each node send  $L_k^p$  to the coordinator node, where all the large  $k$ -sequences  $\mathcal{L}_k := \bigcup_p L_k^p$  are derived.

4. If  $\mathcal{L}_k$  is empty, the algorithm terminates. Otherwise the coordinator node broadcasts  $\mathcal{L}_k$  to all the nodes,  $k := k + 1$  and goto “1”.

*Example 1.* Assuming there are 3 nodes in a system. Let the large items, computed pass 1, be  $L_1 = \{1, 2, 3, 5, 7\}$ . Since SPSPM partitions the candidate sequences in round-robin manner, the candidate sequences are assigned shown in Figure 3. Suppose Node 0 reads the customer-sequence  $s = \{1, 3, 6\}\{2, 4\}$ . Then Node 0 broadcast  $s$  to all the other nodes and increment the count-support of  $\{1, 3\}, \{3\}\{2\}$ . Node 1 and Node 2 receive the sequence  $\{1, 3, 6\}\{2, 4\}$  from Node 0. Node 2 increments the support-count of  $\{1\}\{2\}$ .

L1={1, 2, 3, 5, 7}

| Node 0 | Node 1 | Node 2 |
|--------|--------|--------|
| {1}{1} | {1,2}  | {1}{2} |
| {2}{3} | {2,5}  | {2}{5} |
| {5}{1} | {5}{2} | {5}{3} |
| {1,3}  | {1}{3} | {1,5}  |
| {2,7}  | {2}{7} | {3}{1} |
| {5}{5} | {5,7}  | {5}{7} |
| {1}{5} | {1,7}  | {1}{7} |
| {3}{2} | {3}{3} | {3,5}  |
| {7}{1} | {7}{2} | {7}{3} |
| {2}{1} | {2}{2} | {2,3}  |
| {3}{5} | {3,7}  | {3}{7} |
| {7}{5} | {7}{7} |        |

Node 0 : Read Customer-sequence {1,3,6}{2,4}

Count {1,3}, {3}{2}

Broadcast {1,3,6}{2,4} to all the other nodes



Node 1 : Receive {1,3,6}{2,4}

Node 2 : Receive {1,3,6}{2,4} → Count {1}{2}

**Fig. 3.** An example of SPSPM algorithm

### 3.3 Hash Partitioned Sequential Pattern Mining : HPSPM

In SPSPM, the amount of communications becomes very large, since SPSPM requires broadcasting all the customer-sequences in the count support processing. HPSPM partitions the candidate sequences among the nodes using the hash function like in the hash join in relational database systems, which eliminates the need for customer-sequence broadcasting and can reduce the comparison workload significantly. Figure 4 gives the behavior of pass  $k$  by the  $p$ -th node, using the notation in Table 2.

```

 $k \geq 2$
 \mathcal{C}_k := The candidates of size k generated from \mathcal{L}_{k-1}
 $\{C_k^p\}$:= All the candidate k -itemsets, whose hashed value corresponding to the p -th
 node
forall $s \in \mathcal{D}^p$ do
 $s' :=$ Select all the items that contained in \mathcal{C}_k from the customer-sequence s
 forall k -sequence $x \in s'$ do
 Determine the destination node ID by applying the same hash function which
 is used in candidate sequences partitioning, and send that k -sequences to it.
 If it is its own ID, increment the support-count for the sequence.
 Receive k -sequences from the other nodes and increment the support-count
 for that sequence
 end
end
 $\{L_k^p\}$:= All the candidates in C_k^p with minimum support
Send L_k^p to the coordinator node
/* Coordinator node make up $\mathcal{L}_k := \bigcup_p L_k^p$ and broadcast to all the nodes */
Receive \mathcal{L}_k from the coordinator node

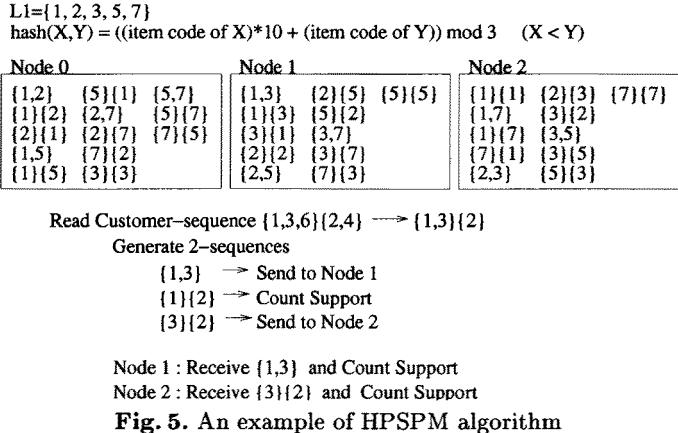
```

**Fig. 4.** HPSPM algorithm

Each node works as follows:

1. Generate the candidate sequence:
  - Each node generates  $\mathcal{C}_k$  using  $\mathcal{L}_{k-1}$ .
  - Apply the hash function to the candidates in  $\mathcal{C}_k$  and determine the destination node ID. If the ID is its own, insert it into the candidate hash table ( $C_k^p$ ).
2. Scan the customer-sequence database and count the support-count value:
  - Each node reads the customer-sequence database from its local disk, and generates new customer-sequence  $s'$  by selecting all the items that are contained in  $\mathcal{C}_k$  from the customer-sequence  $s$ .
  - Generate  $k$ -sequences from  $s'$  and apply the same hash function used in phase 1. Derive the destination node ID and send the  $k$ -sequence to it.
  - For the sequences received from other nodes and those locally generated whose ID equals the node's own ID, search the candidate hash table. If hit, increment its support-count value.
3. – 4. Same as SPSPM

In Figure 5, Example 2 illustrates the behavior of HPSPM for the same condition at Example 1.



*Example 2.* Let the hash function be  $\text{hash}(X, Y) = ((\text{item code of } X) * 10 + (\text{item code of } Y)) \bmod 3$ , here  $X < Y$ . Therefor, the candidate sequence {1, 2} whose hashed value is 0 is allocated to Node 0. The other candidate sequences are allocated the same way. Suppose Node 0 reads the customer-sequence  $s = \{1, 3, 6\}{2, 4}$ . Then Node 0 generates new customer-sequence  $s' = \{1, 3\}{2}$  by selecting the large items in  $s$ , generates 2-sequences from  $s'$  and applies the hash function used at the candidate sequence partitioning. For example, since the hash value of {3}{2} is 2, Node 0 sends {3}{2} to Node 2. Node 2 receives {3}{2} from Node 0 and increments the support-count of {3}{2}.

## 4 Performance Evaluation

We measured all the experiments on a 16-node IBM SP2, which employs a shared-nothing architecture. Each node is a POWER2 with 256MB local memory and a 2GB local disk drive. Each node communicates with each other through HPS (High-Performance Switch).

To evaluate the performance of the proposed parallel algorithms, synthetic datasets emulating retail transactions are used, where the generation procedure is based on the method described in [2]. Table 3 shows the meaning of the various parameters and the characteristics of the datasets used in our experiments. In the following, we use the notation  $CxTy$  to denote a dataset in which an average number of transactions per customer-sequence is  $x$  and an average number of items per transaction is  $y$ .

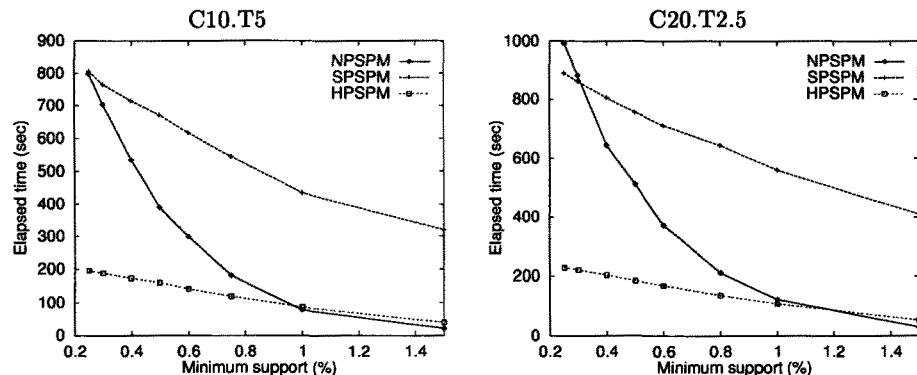
### 4.1 Measurement of Execution Time

Figure 6 shows the execution time of the proposed parallel algorithms for pass 2, varying the minimum support condition. The customer-sequence database is evenly spread over the node's local disks before the experiments starts. In these

**Table 3.** Parameters of datasets

| Parameter                                                          | C10T5   | C20T2.5 |
|--------------------------------------------------------------------|---------|---------|
| Number of customer-sequences (= size of Database)                  | 384,000 | 384,000 |
| Average number of transactions per customer-sequence               | 10      | 20      |
| Average number of items per transaction                            | 5       | 2.5     |
| Average length of maximal potentially frequent sequences           | 4       | 4       |
| Average size of Itemsets in maximal potentially frequent sequences | 1.25    | 1.25    |
| Number of maximal potentially frequent sequences                   | 5,000   | 5,000   |
| Number of maximal potentially frequent itemsets                    | 25,000  | 25,000  |
| Number of items                                                    | 10,000  | 10,000  |

experiment, we measure the performance only for pass 2. Since the single node's memory cannot hold the entire candidate sequences only for pass 2.

**Fig. 6.** Execution time for pass 2

In NPSPM, the execution time increases sharply when minimum support becomes small. When the minimum support is small, the size of the candidate sequences becomes large and the single node's memory cannot hold the entire candidate sequences. In such case, NPSPM divides the candidate sequences into fragments, and scans the customer-sequence database repetitively for each fragment. Thus the execution time of NPSPM increases sharply as decreases the minimum support threshold.

HPSPM always outperforms SPSPM. In HPSPM and SPSPM, each node exchanges the customer-sequence data among the nodes for support count processing, since these two algorithms partition the candidate sequences over the memory space of all the nodes. Since SPSPM partition the candidate sequences arbitrarily among the nodes, each node has to broadcast its local customer-sequence to all the other nodes, which causes a large amount of communications. On the other hand, since HPSPM partitions the candidate sequences using hash function, each node transmits the sequences which are generated from the items in a customer-sequence to certain node, the communication costs is reduced.

## 4.2 Speedup

Figure 7 shows the speedup ratio for pass 2. Here, the number of nodes are varying 4, 6, 8, 12 and 16. The curve is normalized with the execution time on 4 nodes. The minimum support threshold was set to 0.75% for C10T5 and 0.8% for C20T2.5.

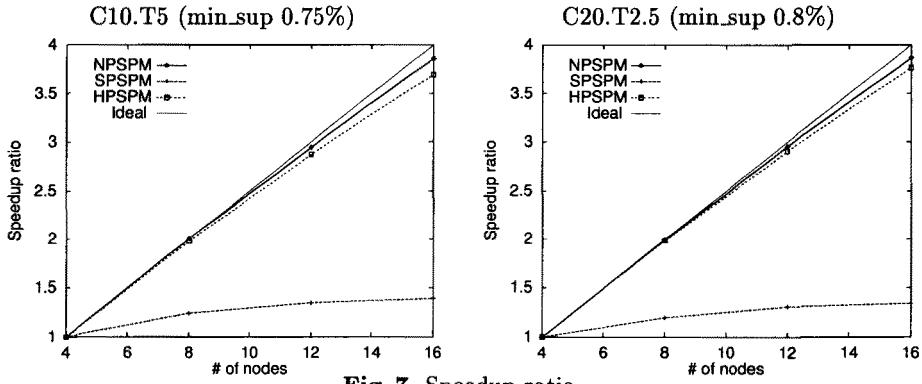


Fig. 7. Speedup ratio

HPSPM and NPSPM attain much higher linearity than SPSPM. Since SPSPM has to broadcast the customer-sequence data to all the other nodes, the communication cost of SPSPM is very high. On the other hand, HPSPM transmits the sequences to certain node, HPSPM can reduce the communication cost. Since NPSPM requires no customer-sequence data transfer for count support processing, NPSPM can attain good linearity. However, the candidate sequences are divided and the support-counts are counted by scanning the customer-sequence database repeatedly in the case the entire candidate sequences do not fit within the memory space of a single node, and each node works on the entire candidate sequences. As a result, the response time of NPSPM is long. Figure 7 show that HPSPM can attain comparable good linearity.

## 5 Conclusions

In this paper, we proposed three different parallel algorithms for mining sequential patterns on a shared-nothing parallel machine. Since mining sequential patterns requires multiple passes over the customer-sequence database and the database to be mined are often very large, parallel algorithms are required.

In NPSPM, the candidate sequences are just copied over all the nodes, no customer-sequence data transfer for count support processing. However, if the entire candidate sequences do not fit with in the memory space of a single node, the candidate sequences are divided to fragments and the support-counts are counted by scanning the customer-sequence database repeatedly. Since the size of the customer-sequence database is very large, it is very costly to scan the database repeatedly for each fragment. The remaining two algorithms, SPSPM and HPSPM, not only partition the customer-sequence database but partition the candidate sequences among all the nodes, which can exploit the total system's memory effectively, and count the support-count of the candidate sequences by

transmit the customer-sequence data. Since SPSPM partition arbitrarily partition the candidate sequences among the nodes, each node has to broadcast its local customer-sequence data to every other nodes, a large amount of communications is caused. HPSPM partitions the candidate sequences using hash function, which avoids the customer-sequence data broadcasting. Consequently, the comparison system workload is reduced significantly. We evaluated the performance of the three algorithms by implementing them on 16-nodes shared-nothing parallel machine, and showed HPSPM can attain relatively good performance.

In [2], generalized sequential pattern is introduced. Since generalized sequential patterns include the items across different levels of the taxonomy over the data items, we have to examine the associations between the different levels of the taxonomy in such mining. Thus, the size of the candidate sequences becomes much larger than mining naive single level sequential patterns. Extension of our parallel algorithms to the mining of generalized sequential pattern is interesting study for future work.

## Acknowledgment

This work is (partially) supported as the Grant-in-Aid for Creative Basic Research #09NP1401: "Research on Multimedia Mediation Mechanism for Realization of Human-oriented Information Environments" by the Ministry of Education, Science, Sports and Culture.

## References

1. R.Agrawal and R.Srikant. Mining sequential patterns. In *Proc. of the 11th Int. Conf. on Data Engineering*, pages 3–14, March 1995.
2. R.Srikant and R.Agrawal. Mining sequential patterns: Generalizations and performance improvements. In *Proc. of the 5th Int. Conf. on Extending Database Technology*, March 1996.
3. S.-J.Yen and A.L.P.Chen. An efficient approach to discovering knowledge from large databases. In *Proc. of 4th Int. Conf. on Parallel and Distributed Information Systems*, pages 8–18, December 1996.
4. T.Shintani and M.Kitsuregawa. Hash based parallel algorithms for mining association rules. In *Proc. of 4th Int. Conf. on Parallel and Distributed Information Systems*, pages 19–30, December 1996.
5. J.S.Park, M.-S.Chen, and P.S.Yu. Efficient parallel data mining for association rules. In *Proc. of the 4th Conf. on Information and Knowledge Management*, pages 31–36, November 1995.
6. D.W.Cheung, J.Han, V.T.Ng, A.W.Fu, and Y.Fu. A fast distributed algorithms for mining association rules. In *Proc. of 4th Int. Conf. on Parallel and Distributed Information Systems*, pages 31–42, December 1996.
7. R.Agrawal and J.C.Shafer. Parallel mining of association rules. In *IEEE Trans. on Knowledge and Data Engineering*, Vol.8, No.6, pages 962–969, December 1996.
8. D.W.Cheung, V.T.Ng, A.W.Fu, and Y.Fu. Efficient mining of association rules in distributed databases. In *IEEE Trans. on Knowledge and Data Engineering*, Vol.8, No.6, pages 911–922, December 1996.
9. E.-H.(Sam)Han, G.Karypis, and V.Kumar. Scalable parallel data mining for association rules. In *Proc. of 1997 ACM SIGMOD Int. Conf. on Management of Data*, pages 277–288, 1997.

# Wavelet Transform in Similarity Paradigm

Zbigniew R. Struzik, Arno Siebes

Centre for Mathematics and Computer Science (CWI)  
Kruislaan 413, 1098 SJ Amsterdam  
The Netherlands  
*email:* Zbigniew.Struzik@cwi.nl

**Abstract.** Searching for similarity in time series finds still broader applications in data mining. However, due to the very broad spectrum of data involved, there is no possibility of defining one single notion of similarity suitable to serve all applications.

We present a powerful framework based on wavelet decomposition, which allows designing and implementing a variety of criteria for the evaluation of similarity between time series. As an example, two main classes of similarity measures are considered. One is the global, statistical similarity, which uses the wavelet transform derived Hurst exponent to classify time series according to their global scaling properties. The second measure estimates similarity locally using the scale-position bifurcation representation.

## 1 Introduction

Many data mining algorithms exist for (more or less) standard, relational data, see, e.g. [1]. However, in practice there is much non-relational data. The most important example is time series data. For example, banks have standard data on their clients, e.g. their names, where they live et cetera, but also a time series giving the status of their account over time.

To use existing data mining technology on such data means that the time series data has to be reduced to a fixed number of characteristics. A very simple idea would be to use the current status of the account as an extra field in the table. However, if we are going to use the data for credit scoring, the current status of the account is likely to miss out on important information. For example, two clients  $A$  and  $B$  could have both \$10.000 in their account now, which is the normal status for  $A$ , whereas it is a one-time record for  $B$ . In such a case, the credit rating would be (much) higher for  $A$  than for  $B$ .

In other words, the behaviour of a time series over time is among the important characteristics of that time series. This means that we have to represent the behaviour of a time series with a finite number of characteristics. Of course, this representation should be such that two time series which show similar behaviour should be close to one another in the representation space, and vice versa.

Crucial in this statement is, of course, what similarity actually means. The precise meaning of similarity is strongly dependent on the intended usage of the representation. Sometimes the trend of the series is the important factor in

determining similarity, whereas in other cases, it is everything but the trend. Sometimes it is global (statistical) behaviour which is important, whereas in other cases it is highly localised behaviour.

It is, therefore, not possible to define one specific measure of similarity between time series that is useful under all circumstances. Rather, a flexible tool-box, in which the user can indicate what is important in this specific case, is necessary. In this paper, we introduce a framework for such similarity measures based on wavelets. Moreover, we study the two extremes in this spectrum of possibilities in depth. The framework for similarity as developed in this paper is based on the fractal analysis of time series [2, 3, 4]<sup>1</sup>.

The topic of the similarity of time series for data mining is not new. Important papers in this area are [5] and [6]. As an aside, note that these papers have other motives in showing why this topic is important. The most important difference between [5] and [6] on the one hand and our paper on the other is the framework. The core criterium for similarity used in [5], *de facto* requires *a priori* determining of what the time series is and what are the outliers or noise. Only then can the actual distance,  $\epsilon$  sausage criterium, work.

Earlier work by the same authors [7] suggested matching in the space of the Discrete Fourier Transform representation. However, DFT in itself provides only global information. Moreover, as is also concluded in [5], this approach fails in the presence of linear bias and is rather sensitive to local outliers.

The work reported in [6] is based on local transformations of the time series. Since there is a choice of the allowed set of transformations, this approach is closer to our approach. Our approach, however, does not rely on an (implicit) underlying model, and thus is not sensitive to outliers, noise, and translations of the data.

In other words, we are not building our similarity framework using a particular similarity model. Rather we utilise a flexible hierarchical representation of a time series (up to a certain resolution). This representation in turn can be tailored to fit matching criteria required. In this approach, one can build in insensitivity to factors other than matching criteria, these often being outliers, noise, translation, scaling or polynomial bias.

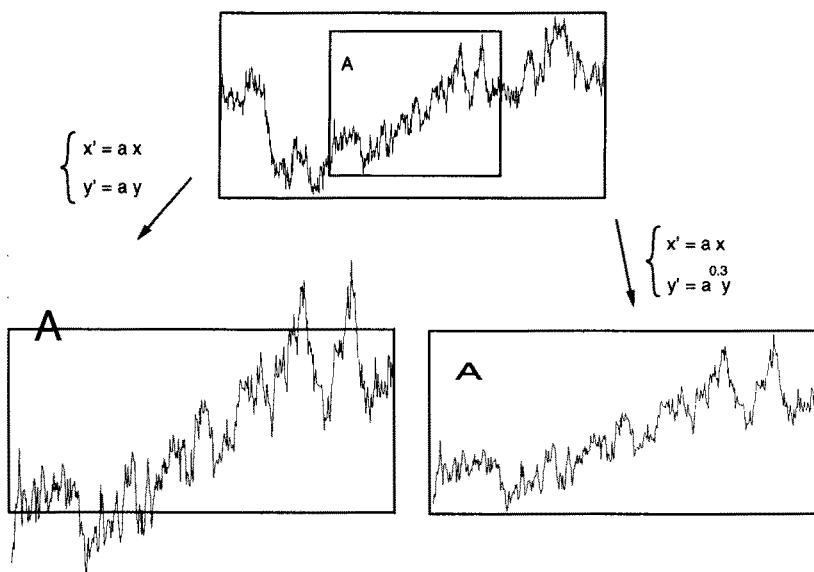
We will continue with the discussion of similarity in the next section, introducing appropriate exponents for both global and local characterisation. In section 3, we will introduce the Wavelet Transform with the appropriate representations. We will elaborate on the global and local similarity measures in sections 4 and 5 respectively. In both sections examples we will give account for some of the most powerful abilities of the methods. Finally, a closing word will be given in section 6.

---

<sup>1</sup> In this paper we refer to *time series* rather than *signals*. However, the term ‘signals’ is exclusively used in the signal processing literature to which reference is made. Both mean one and the same thing - the sample (not necessarily uniform) record of data.

## 2 Global Versus Local Similarity

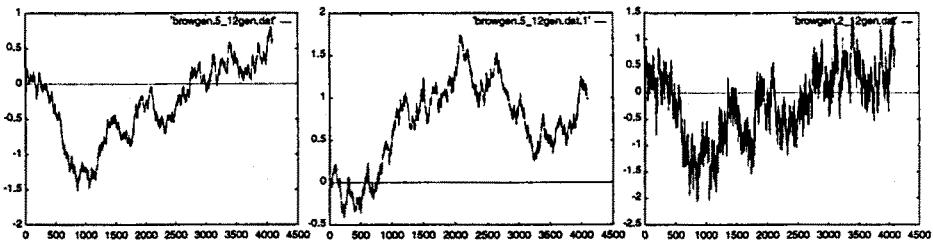
The global parameter which we would like to use for characterising (the roughness of) the time series should not change if we estimate it for the first or the second half or any arbitrary part of the time series, provided the characteristics of the time series do not change in time (stationarity) or with the length of the sample. The former requires that parameters remain stable with respect to scaling within a considerable range of scales (scaling, self-affinity). A good parameter indication of the similarity of the time series with its parts is the exponent with which one has to re-scale the height of the (sliding) window with the part of the time series in order to obtain a time series similar to the one compared. In Figure 1 below, we illustrate this concept for the case of part of the time series compared to the complete time series itself.



**Fig. 1.** Shows the horizontal versus vertical rescaling argument that the exponent characterises the time series globally. *Similar* rescaling in the bottom left figure versus *affine* rescaling, bottom right, of the fractional Brownian motion of  $H = 0.3$ . The rescaling factor used for the affine rescaling of the  $(x, y)$  axis is  $(a, a^{0.3})$ , while for the similar case both axes were rescaled using the  $a$  factor.

This concept of self-affinity and the related Hurst exponent  $H$  has been developed within the domain of fractal geometry and is broadly applicable for time series from sources in both natural and computer sciences. In particular, it can be shown that the exponent  $H = 0.5$  corresponds to the Brownian path (or trail, or motion) - a random process with independent increments - the integral

of random noise.  $H > 0.5$  is the evidence of a long range positive correlation in the time series, visually effecting a time series with tempered jumps. On the contrary,  $H < 0.5$  gives evidence of a negative correlation, a so-called anti-correlation, which is displayed by more ‘wild’ behaviour. These processes are commonly referred to as the fractional Brownian motion (fBm).



**Fig. 2.** Example time series,  $H = 0.5$  for first two,  $H = 0.2$  for the most right. Only the first two from the left are statistically similar. On the other hand, the first and the third time series are almost identical if difference in scaling is neglected.

As a global measure,  $H$  can be successfully used to compare time series which are *statistically* similar, provided it is estimated in a trustworthy manner (including the removal of non-stationarities in the time series and providing the limits on scaling range and the realistic error bounds). For this purpose, we will use the wavelet transform which has been shown to be a particularly successful tool in assessing the scaling behaviour of time series [3].

Two arguments for the generalisation (of the global notion of similarity) suggest themselves; one is that we could allow the Hurst exponent to vary with position or we could be interested in local rather than global similarities between time series. Both require making the characteristics of the time series local in position. The relevant concept is known as the Hölder exponent  $h$  of the function in  $x_0$  - if there exists a polynomial  $P_n(x)$  of the degree  $n$  such that:

$$|f(x) - P_n(x - x_0)| \leq C|x - x_0|^h, \quad (1)$$

then  $h$  is said to be the local Hölder exponent of the function and it characterises the scaling of the function locally for  $n < h \leq n + 1$ . The polynomial  $P_n$  corresponds to the Taylor series expansion of  $f$  around  $x_0$  up to the order  $n$ .

The Wavelet Transform (which we will describe below) has been demonstrated to be a tool exceptionally well suited to the estimation of this exponent and in fact, as we will see later, global estimates like the Hurst exponent are obtained through this local Hölder exponent by means of taking an ensemble average in an appropriate partition function.

In conclusion, we have been able to identify two major approaches to stating similarity of time series; the global (statistical) similarity and the local, feature based similarity. In the following we will demonstrate how to approach both

classes with a common formalism based on the wavelet transform decomposition of time series.

### 3 Continuous Wavelet Transform and its Maxima Used to Reveal the Structure of the Time Series

As already mentioned above, the recently introduced Wavelet Transform (WT), see e.g. Ref. [8], provides a way of analysing local behaviour of functions. In this, it fundamentally differs from global transforms like the Fourier Transform. In addition to locality, it possesses the often very desirable ability of filtering the polynomial behaviour of some predefined degree.

Conceptually, the wavelet transform is a convolution product of the time series with the scaled and translated kernel - the wavelet  $\psi(x)$ <sup>2</sup>.

$$(Wf)(s, b) = \frac{1}{s} \int dx f(x) U(s, b) \psi(x). \quad (2)$$

The scaling and translation actions are incorporated as the operator  $U(s, b)$ ; the scale parameter  $s$  ‘adapts’ the width of the wavelet kernel to the *microscopic resolution* required, thus changing its frequency contents, and the location of the analysing wavelet is determined by the parameter  $b$

$$U(s, b)\psi(x) = \psi\left(\frac{x-b}{s}\right), \quad (3)$$

where  $s, b \in \mathbf{R}$  and  $s > 0$  for the continuous version (CWT).

Figure 3 shows how the wavelet transform reveals more and more detail while going towards smaller scales. The wavelet transform is sometimes referred to as the ‘mathematical microscope’, due to its ability to focus on weak transient frequencies and singularities in the time series. The wavelet used determines the optics of the microscope; its magnification varies with the scale factor  $s$ .

The analysis of the local singular properties of a function (usually it is the singularities and not the smooth behaviour which are interesting in the time series) with the wavelet transform can be illustrated by the following. As already mentioned, the singularity strength is often characterised by the so-called Hölder exponent, compare Eq.1.

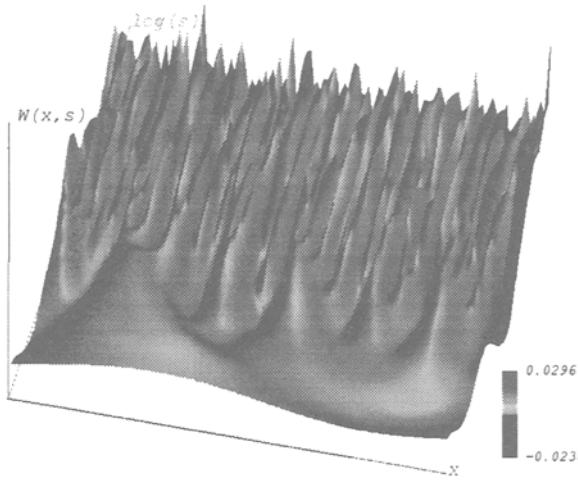
If we represent the function  $f$  through its Taylor expansion around  $x = x_0$ :

$$f(x)_{x_0} = c_0 + c_1(x - x_0) + \dots + c_n(x - x_0)^n + C|x - x_0|^{h(x_0)}.$$

It follows directly that if  $h(x_0)$  is equal to a positive integer  $n$ , the function  $f$  is  $n$  times continuously differentiable in  $x_0$ . Alternatively, if  $n < h(x_0) < n + 1$  the function  $f$  is continuous and singular in  $x_0$ . In that case  $f$  is  $n$  times differentiable, but its  $n^{\text{th}}$  derivative is singular in  $x_0$  and the exponent  $h$  characterises

---

<sup>2</sup> The power given to the normalising factor  $s$ , it is often chosen to serve a particular purpose. In this work, we choose a default factor  $s^{-1}$ , which conserves the integral  $\int dx |\psi(x)|$  and thus leaves the  $L^1$  measure invariant.



**Fig. 3.** WT representation of the time series from Figure 2 leftmost. The wavelet used is the Mexican hat.

this singularity. The exponent  $h$ , therefore, gives an indication of how regular the function  $f$  is in  $x_0$ , that is the higher the  $h$ , the more regular the function  $f$ .

The wavelet transform of the function  $f$  in  $x = x_0$  with the wavelet of at least  $n$  vanishing moments, i.e. orthogonal to polynomials up to (maximum possible) degree  $n$ :

$$\int_{-\infty}^{+\infty} x^m \psi(x) dx = 0 \quad \forall m, 0 \leq m < n,$$

reduces to

$$W^{(n)} f(s, x_0) \sim C \int \psi(x) |s x|^{h(x_0)} dx \sim C |s|^{h(x_0)} \int \psi(x') |x'|^{h(x_0)} dx'.$$

Therefore, we have the following proportionality of the wavelet transform of the singularity  $n \leq h \leq n + 1$ , with the wavelet with  $n$  vanishing moments:

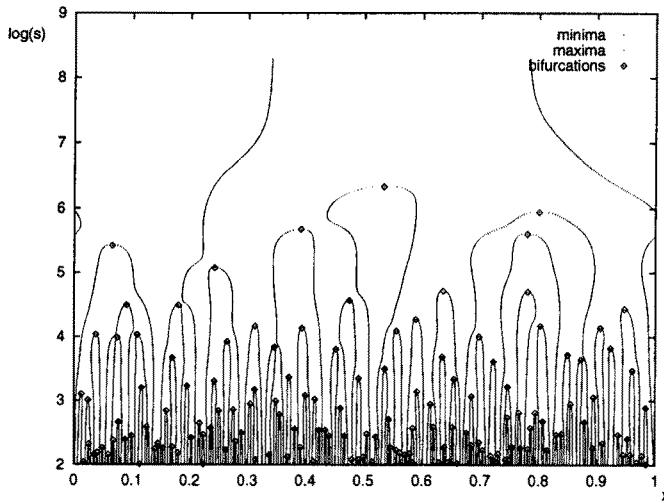
$$W^{(n)} f(s, x_0) \sim |s|^{h(x_0)}.$$

Thus the continuous wavelet transform can be used for detecting and representing the singularities in the time series even if masked by the polynomial bias. This ability is inherited by the more efficient representation based on modulus maxima of CWT, on which we will shortly expand in the following.

### 3.1 Wavelet Transform Modulus Maxima Representation and Bifurcation Representation

The continuous wavelet transform described in Eq. 2 is an extremely redundant representation. Therefore, other, less redundant representations, are frequently

used, including orthogonal representations and a variety of frames (almost orthogonal representations) [8]. For our purpose of comparison of features in the time series, one critical requirement is the translation shift invariance of the representation; no other than the boundary coefficients of the representation should change, if the time series is translated by some  $\Delta t$ . A useful representation satisfying this requirement and of much less redundancy than the CWT is the Wavelet Transform Modulus Maxima (WTMM) representation, introduced by Mallat [9]. It is derived from the CWT representation by extracting lines of maxima of the modulus of the wavelet transform. An example WTMM tree is shown in Figure 4, together with the highlighted bifurcations of the maxima lines [4].



**Fig. 4.** WTMM representation of the time series and the bifurcations of the WTMM tree.

#### 4 A Global (Statistical) Estimation of the Similarity of Time Series in Presence of Scaling, Translation and Polynomial Bias

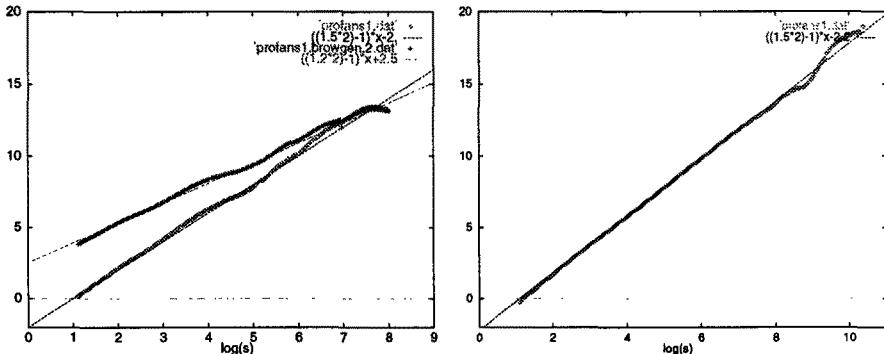
It has been shown by Arneodo et al [3], that the WTMM tree is particularly useful for estimating the scaling parameters of functions. In particular, the Hurst exponent is related to the  $q = 2$ nd moment (correlation) of the scaling of the measure on the WTMM maxima tree:

$$s^{2(H+P_n)-1} \sim \sum_{\text{all maxima at scale } s} \mu^2(s) \quad (4)$$

where  $\mu(s)$  is the amplitude of the maximum of the WT at the corresponding scale, and the sum - the partition function - is taken over all the maxima at the given scale  $s$ .  $P_n$  indicates the degree of the polynomial offset of the time series.

This relation (for  $P_n = 1$ ) can be easily verified in Figure 5 where in log-log coordinates the power law relation 4 should result in a straight line.

We show the same, second moment for two examples of random noise and anti-correlated fractional noise. The Hurst exponent can easily be estimated from the slope of the linear fit to the scaling portion of the plot.

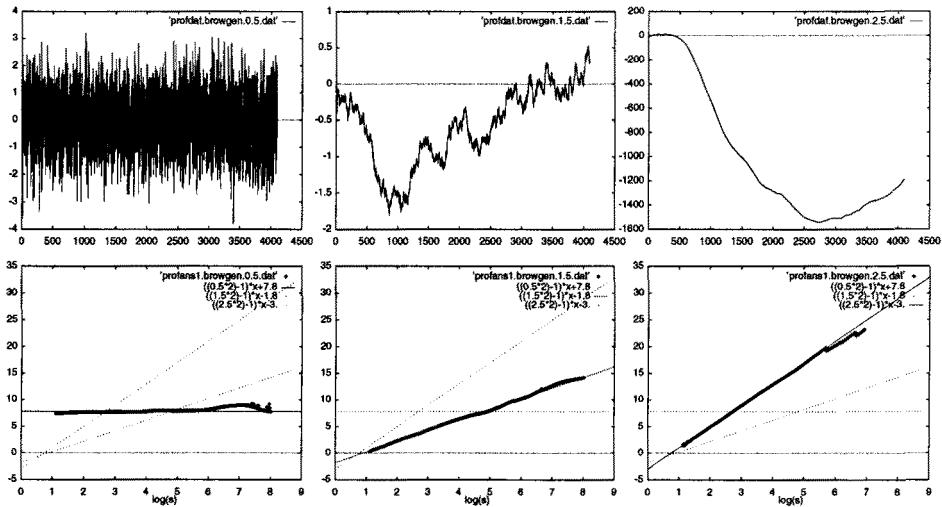


**Fig. 5.** Left: the second moments of the modulus maxima representation of the noise samples from Figure 2 leftmost and rightmost. Right: the same for the real life sample of a financial index shown in Figure 11 left.

In Figure 5 right, we show the same second moment estimated for the record of the financial index. It falls very well into the same category as the simulated Brownian path - indeed financial records are known to follow the  $H=0.5$  law very closely, see e.g. [10]. The exponent  $H$  can thus be used to classify global similarity between time series or categorize them on the statistical grounds.

By design,  $H$  is limited to take values from the interval  $0 \leq H \leq 1$ . With the WTMM based formalism, we are able to estimate not only the fractional scaling part but also the degree of the underlying polynomial  $Pn$ . For fBm trails  $Pn = 1$ , for the noise record it would be  $Pn = 0$ . It is therefore more convenient to take the complete exponent  $\beta = H + Pn$  as the (correlation) exponent representative to our time series.

With this exponent, we are able to distinguish between various categories of time series for  $0 < \beta \leq 1$ ,  $1 < \beta \leq 2$ ,  $2 < \beta \leq 3$  or higher. As an illustration, see Figure 6, where we show the scaling of the WTMM correlation dimension evaluated for random noise, its integral - Brownian trail - and, again, its integral. For each integration step, the increase of the slope of the second moment is two, and the corresponding increase of the correlation dimension is one. Note that WTMM based formalism will correctly estimate the correlation exponent  $\beta$  only if the wavelet used has enough vanishing moments. In most practical situations, this condition is satisfied with  $n = 2$  or  $n = 3$ . For example,  $n = 1$  is enough for noise record  $0 < \beta \leq 1$ , like the leftmost example in Figure 6. But we need a wavelet with at least  $n = 2$  for the  $1 < \beta \leq 2$  class (central example in Figure 6) and with at least  $n = 3$  for the  $2 < \beta \leq 3$  class (rightmost example in Figure 6).



**Fig. 6.** Above, from the left: the random noise sample, its integral - Brownian trail and again its integral. Below from the left: corresponding scaling of the second moments of WTMM maxima.

## 5 Local Similarity Estimation in the Presence of Scaling, Translation and Polynomial Bias

In order to evaluate the local similarity, we will turn to local features of the time series, the bifurcations of the WTMM tree. Bifurcations [4] form a set of highly sensitive ‘landmarks’ in the WT landscape of the decomposed function. By reflecting the scale-position development of the maxima tree, they capture the singular structure of the time series. Each bifurcation can be represented with its position and scale coordinates, plus the corresponding value of the WT in the bifurcation point.

Just like the wavelet transform itself, the bifurcations can be evaluated for the time series up to a certain resolution, meaning that only the coarse features are taken into account. Alternatively, a range of scales can be determined in the application to be covered by the WTMM tree and its bifurcations. The numbers used for our experiments ranged from 20-100 bifurcations, covering the span of a maximum of two decades of scales from the highest resolution available.

We will use bifurcations obtained from the WTMM tree of the wavelet transform of the time series with the Mexican hat wavelet, the second derivative of the Gaussian kernel. This means that the maxima lines follow singular features in the second derivative of the function and the bifurcation representation reflects the structure of these features. This will allow for looking for similarities with respect to linear bias - such bias is filtered out from the time series by the wavelet with two vanishing moments. One can, of course, use wavelets with fewer or more vanishing moments to suit one’s particular needs.

## 5.1 Local Distance (Similarity) Measure between Two (or More) Bifurcations

Essentially, the method uses the bifurcation representations of two time series to be compared and estimates the degree of similarity of these representations. Additionally one can shift both representations with respect to one another in order to find whether there is a better match if shift and scaling are involved. Note that a shift in the logarithmic scale corresponds with the scaling operation in the original time series.

The simplest but quite reliable measure of the similarity of two sets of bifurcations is given by the occurrence of a bifurcation in one representation, within a distance  $\epsilon$  of some ‘reference’ bifurcation in the ‘reference’ set of bifurcations. Counting the fraction of reference bifurcations which have such a matching counterpart in the bifurcation set compared gives the estimate of the similarity between the two sets - a number from the range 0..1. A useful improvement of this scheme is easily made by counting only bifurcations in which WT has the same sign. This procedure, subject to one  $\epsilon$  parameter only, gives good results.

A straightforward extension to the box of  $\epsilon$  size is a two-dimensional correlation function which has a smooth decay of the distance between the bifurcations. As the measure of correlation between two bifurcation points:  $biff_a(a, \sigma_b)$  and  $biff_b(b, \sigma_b)$ , we took for our experiments the auto-correlation function of two Gaussian kernels  $C(biff_a, biff_b)$ . This correlation can be parametrized by the additional shift in (logarithmic) scale and position  $\Delta s$  and  $\Delta x$  respectively, see [11] for more details.

With this measure for the correlation of two (or more) bifurcations, we can now estimate the total correlation of two bifurcation representations of the time series we want to compare. The most straightforward measure is simply

$$M(\Delta x, \Delta s) = \frac{1}{norm(N_1, N_2, \Delta x, \Delta s)} \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} C(biff_i, biff_j)(\Delta x, \Delta s), \quad (5)$$

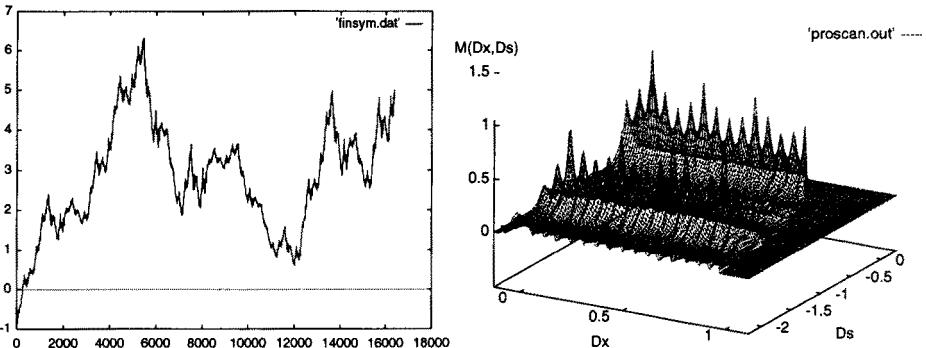
where  $N_1$  and  $N_2$  are the respective numbers of bifurcations in both compared time series representations. Generally,  $norm$  should also be an appropriate function of the scale and position shift in order to compensate for the change in scale-position overlap of compared bifurcation representations. For more details on the measure and algorithm design, the reader can refer to [11].

In the rest of this section we will demonstrate the ability of the method to localise similarity in time series using the measure just designed, Eq. 5, to compare their intricate structure - the scale-position behaviour of the second derivative of the time series. This structure is captured by the bifurcation representation obtained with the wavelet orthogonal to linear bias ( $n = 2$ ), which we will use in all the examples in the rest of this paper. It is, of course, possible to use a structure of a different derivative of the time series or the time series itself by taking a wavelet with an appropriate  $n$  - number of vanishing moments.

## 5.2 The Effect of Translation and Scaling

In this example, we demonstrate how similarities can be found for the time series which is scaled and translated. Let us take the example time series record, see Figure 7.

Using this example, we will demonstrate that the algorithm using the measure 5 to compare two bifurcation representations is capable of finding similarities between the time series (or their parts), with respect to the operations of translation and scaling.



**Fig. 7.** Example fractal time series consisting of four self-affine sub-parts (left). The measure shows a high regularity and reveals the scaling and translation - elements of the invariance of the time series.

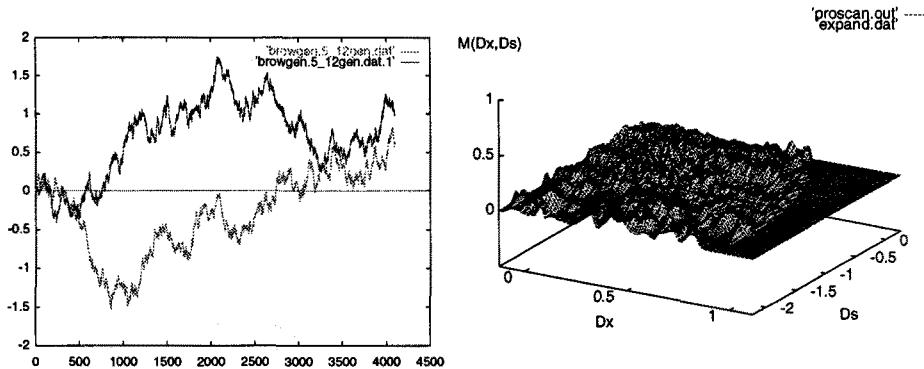
Instead of comparing one time series with another, in this test we will actually compare the time series with itself. Covering an adequate scope of values  $\Delta x, \Delta s$  reveals in the measure  $M(\Delta x, \Delta s)$  the presence of a very structured self-affinity relation within this time series - there are four main peaks located at  $\Delta x = 0, \Delta x = 1/4, \Delta x = 1/2$  and  $\Delta x = 3/4$ . In between these, there are lower peaks, starting from the big peak at  $\Delta x = 0$ . The next smaller peak is at  $\Delta x = 1/16$ , followed by another at  $\Delta x = 2/16$ , and the next at  $\Delta x = 3/16$ . This sequence is repeated at  $\Delta s = -1.39 = \log(1/4)$ .

This, in fact, goes to show that we discovered that within the time series there are four similar parts, which in sequel contain four similar parts, etc. This similarity is evaluated with respect to the second derivative of the time series - the analysing wavelet is the Mexican hat - all the masking linear trend at different scales has been removed. Indeed, the test time series is an IFS fractal [12] with four non-overlapping self-affine transformations as the construction rule.

## 5.3 The Effect of Random Noise

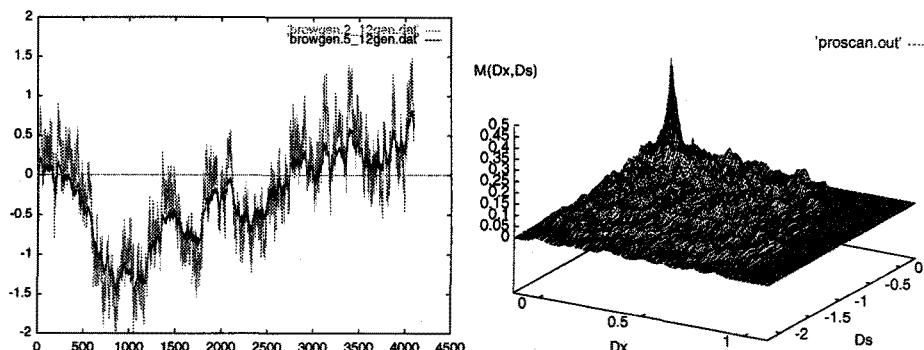
Speaking of the influence of random noise has, of course, a very special meaning in the context of our analysis - most of the example test time series we considered

were records of pure or correlated noise. Still such noise is a perfectly valid and valuable time series (in fact a record which in absence of an appropriate model seems to be just an uncorrelated noise is likely to contain perfectly coded information). Therefore, in addition to comparing two different noise records, it makes perfect sense to evaluate how one noise record is corrupted by other noise. We took three examples.



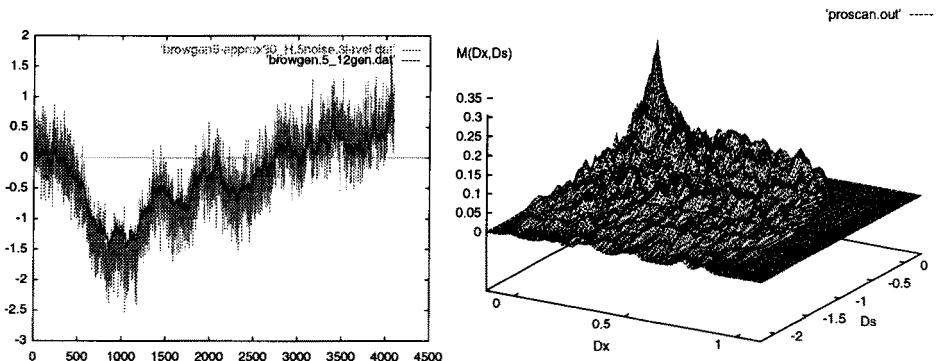
**Fig. 8.** Two independent random walks (left) investigated for the presence of similar parts with respect to scaling translation and linear bias show only slight local similarities at a residual level of measure fluctuations (right).

In the first, Figure 8, two independent random walks are scanned for similarities. The level of the measure remains low but significant within the searched range of scale and position shift. It can be considered as the fluctuations of the similarity measure reaching a significant level due to random occurrences of parts remotely looking like one another. These similarities are more likely to be assumed when going to higher  $\Delta x, \Delta s$ , due to the simple fact of considering the overlap of just a few bifurcations[11].



**Fig. 9.** Two noises with  $H = 0.5$  and with  $H = 0.2$  created with the same random sequence. Right, the similarity measure response of about 0.5 at no shift.

In the second example, figure 9, we consider two noises created with the same random sequence. One is uncorrelated with  $H = 0.5$  and the other anti-correlated with  $H = 0.2$ . Both are created with the ‘random midpoint displacement method’, see e.g. [2], using the same random sequence, meaning the sign of displacement is consistent in both samples. Indeed, the similarity measure gives quite a good response, reaching about 0.5 at no shift.



**Fig. 10.** Example random walk with the addition of random noise at  $-10dB$  level. Right the resulting measure.

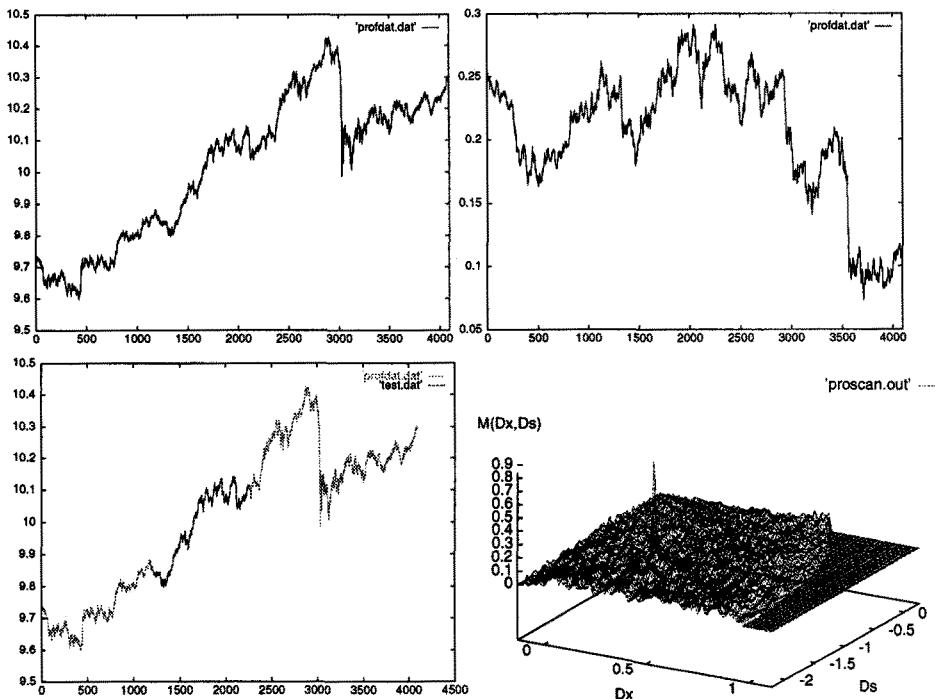
The third example, figure 10, demonstrates the influence of the additive random noise at a level of about  $-10dB$  (amplitude factor 0.3). The result is a drop in the measure associated with a noticeable widening of the maximum cone. The resulting measure is at a level of one third of the measure norm for two identical time series.

#### 5.4 The Real Life Sample within a Sample Example

As the last example, we took a real life time series. Two records of a financial index were scanned for the presence of similar parts with respect to scaling, translation and linear bias.

While from a visual inspection it is rather difficult to establish the degree of similarity between the two, the similarity measure reveals a high degree of similarity at  $\Delta x = 0.3$  and  $\Delta s = -1.4$ . (This corresponds to a shift by  $0.3 \cdot 4096 = 1229$  samples and rescaling by a factor  $e^{-1.4} = 0.25$ )<sup>3</sup>. At almost maximum level  $C = 0.87$ , it is far stronger than the rest of the measure. Applying the shift parameters to the second plot confirms a close fit, but only after the linear trend is restored! (Indeed, the second time series is just a part of the first with added linear bias.)

<sup>3</sup> We added a relatively strong bias in order to illustrate our purpose. The method will, of course, also work for smaller levels of bias for which a visual check of the similarity of both time series will be possible. Still, even in such cases, techniques without the ability to filter polynomial trends are likely to fail.



**Fig. 11.** Above, two example real life time series subject to investigation for the presence of similar parts with respect to scaling, translation and linear bias. Left below, the match revealed in the maximum of the measure shown at right below.

## 6 Final Remarks and Conclusions

Determining similarity of time series is an important problem in data-mining. We presented a powerful framework allowing construction of specific similarity criteria, in the presence of scaling, translation and polynomial bias. Two main classes of similarity evaluation measures were distinguished and the appropriate measures were proposed.

The global, statistical similarity was estimated with the Wavelet Transform derived Hurst exponent. It classifies time series according to their global scaling properties. The local, detail oriented measure, used the scale-position bifurcation representation of the wavelet modulus maxima transform of the time series. It makes it possible to obtain good matches of (the parts of) the time series with respect to scaling translation and polynomial bias. The degree of the polynomial bias filtered can be affected as well as the range of the translation and scaling parameters. The measure used for matching the two bifurcation representations of the time series can also be adapted to the specific user requirements.

This flexibility represents an important difference from existing methods like those presented, for example, in [6, 7, 5]. Let us, in closing, once again stress that the framework presented does not impose a particular model, but facil-

itates implementing models to suit one's needs. We demonstrated this in the two example similarity measures. Since they represent two extremes, it is, of course possible, that both the global and the local measure can be used together with appropriate weighting factors in order to satisfy an intermediate objective. However, the capabilities of the presented framework extend considerably beyond just modifying or merging the example applications presented. Due to the ability to represent the analysed time series in a unique set of position-space localised features (we presented bifurcations but using other features is possible), one can design a variety of (hierarchical) matching algorithms. The features used in such algorithms are dependent on the wavelet used and can inherit all of the advantages presented, such as selective sensitivity to polynomial trends. This is also the direction of current research about which we will be communicating shortly. In addition to this, we will consider issues of increasing the accuracy of the representations with compactly supported wavelets and optimization of the algorithms for speed.

Apart from developing the methodology, our work proceeds towards the particular goal of exploring time series from data-mining applications as described in the introduction.

## References

1. U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, R. Uthurusamy, Eds., *Advances in Knowledge Discovery and Data Mining*, AAAI Press/MIT Press, 1996.
2. K. Falconer, *Fractal Geometry - Mathematical Foundations and Applications*, John Wiley (1990).
3. A. Arneodo, E. Bacry, J.F. Muzy, The Thermodynamics of Fractals Revisited with Wavelets, *Physica A*, **213**, 232-275, (1995).
4. Z.R. Struzik, The Wavelet Transform in The Solution to the Inverse Fractal Problem, *Fractals* **3** No.2, 329-350 (1995).
5. R. Agrawal, K-I. Lin, H.S. Sawhney, K. Shim, Fast Similarity Search in the Presence of Noise, Scaling and Translation in Time-Series Databases, in *Proceedings of the 21 VLDB Conference*, Zürich, 1995.
6. G. Das, D. Gunopulos, H. Mannila, Finding Similar Time Series, In *Principles of Data Mining and Knowledge Discovery*, Lecture Notes in Artificial Intelligence 1263, Springer, 1997.
7. R. Agrawal, C. Faloutsos, A. Swami. Efficient Similarity Search in Sequence Databases, in *Proc. of the Fourth International Conference on Foundations of Data Organization and Algorithms*, Chicago, 1993.
8. I. Daubechies, *Ten Lectures on Wavelets*, S.I.A.M. (1992).
9. S.G. Mallat, S. Zhong, Complete Signal Representation with Multiscale Edges, *IEEE Trans. PAMI* **14**, 710-732 (1992).
10. C.J.G. Evertsz, Fractal Geometry of Financial Time Series, *Fractals* **3** No.3, 609-616 (1995).
11. Z.R. Struzik, A. Siebes, Wavelet Transform in Similarity Paradigm I - CWI Technical Report, *CWI Reports INS-R* 9802, (1998).
12. M.F. Barnsley, *Fractals Everywhere*, Academic Press, NY, (1988).

# Improved Rule Discovery Performance on Uncertainty

Mehmet R. Tolun

Department of Computer Engineering  
Middle East Technical University  
İnönü Bulvarı - Ankara  
Turkey 06531  
[{tolun@ceng.metu.edu.tr}](mailto:{tolun@ceng.metu.edu.tr})

Hayri Sever

Department of Computer Sc. & Eng.  
Hacettepe University  
Beytepe - Ankara  
Turkey 06532  
[{sever@eti.cc.hun.edu.tr}](mailto:{sever@eti.cc.hun.edu.tr})

Mahmut Uludağ

Artificial Intelligence Group  
TÜBİTAK Marmara Research Center  
PK 21 Gebze Kocaeli  
Turkey 41470  
[{mahmut@mam.gov.tr}](mailto:{mahmut@mam.gov.tr})

## Abstract

In this paper we describe the improved version of a novel rule induction algorithm, namely ILA. We first outline the basic algorithm, and then present how the algorithm is enhanced using the new evaluation metric that handles uncertainty in a given data set. In addition to having a faster induction than the original one, we believe that our contribution comes into picture with a new metric that allows users to define their preferences through a penalty factor. We use this penalty factor to tackle with over-fitting bias, which is inherently found in a great many of inductive algorithms. We compare the improved algorithm ILA-2 to a variety of induction algorithms, including ID3, OC1, C4.5, CN2, and ILA. According to our preliminary experimental work, the algorithm appears to be comparable to the well-known algorithms such as CN2 and C4.5 in terms of accuracy and size.

## 1 Introduction

A data-mining process involves extracting valid, previously unknown, potentially useful, and comprehensible patterns from large databases. As described in (Fayyad, 96; Simoudis, 96), this process is typically made up of selection and sampling, preprocessing and cleaning, transformation and reduction, data mining, and evaluation steps. The first step in data-mining process is to select a target data set from a database (or a data warehouse) and to possibly sample the target data. The preprocessing and data cleaning step handles noise and unknown values as well as accounting for missing data fields, time sequence information, and so forth. The data reduction and transformation involves finding relevant features depending on the goal of the task and certain transformations on the data such as converting one type of data to another (e.g., changing nominal values into numeric ones, discretizing continuous values), and/or defining new attributes. In the mining step, the user may apply one or more knowledge discovery techniques on the transformed data to extract valuable patterns. Finally, evaluation step involves interpreting the result (or discovered pattern) with respect to the goal/task at hand. Note that the data-mining process is not linear and involves a variety of feedback loops, because any one step can result in

changes in preceding or succeeding steps. Furthermore, the nature of a real-world data set, which may contain noisy, incomplete, dynamic, redundant, continuos, and missing values, certainly makes all steps critical on the path going from data to knowledge (Deogun et al., 97; Matheus et al., 93).

One of the methods in data mining step is inductive learning that is mostly concerned with the finding of general descriptions of a concept from a set of training examples. Practical data mining tools generally employ a number of inductive learning algorithms. For example, Silicon Graphics' data mining and visualization product MineSet™ uses MLC++ as a base for the induction and classification algorithms (Kohavi et al. 1996). This paper focuses on establishing causal relationships to class labels from values of attributes via a heuristic search that starts with values of individual attributes and continues to consider the pairwise, triple, or further combinations of attribute values in sequence until the example set is covered.

Once a new learning algorithm has been introduced, it is not unusual to see additional development work that modify the algorithm to improve it in various ways (Salzberg, 1995). These improvements are important in establishing a method and clarifying when it is useful. In this paper, we describe two modifications to improve upon such an algorithm, namely Inductive Learning Algorithm- ILA introduced by Tolun and Abu-Soud (1998).

The first modification we performed is the ability to deal with uncertain data. In general, two sources of uncertainty can be distinguished. One of these is noise that may happen because of incorrect recording or transcription of data, or because of incorrect measurement or perception at an earlier stage. The second case is known as incomplete data to point out the fact that some feature values are missing. In real-world problems this often constitutes the greatest source of error, because the fact that data has been organised and collected around the needs of organisational activities causes incomplete data from the viewpoint of the knowledge discovery task. Under such circumstances, the knowledge discovery model should have the capability of providing approximate decisions with some confidence level.

The second modification is the greedy rule generation bias that reduces the learning time by the cost of increased number of generated rules. This feature is discussed more in section 3. ILA-2 is an improved version of ILA with respect to the modifications stated above. In the last section of this paper, using real-world data sets, we empirically compare ILA-2 with ILA as well as with some other well-known inductive algorithms. The results show that ILA-2 is better than ILA in terms of the accuracy in classifying unseen instances, the size of classifiers and the duration of learning process. Test results also show that ILA-2 is comparable to both CN2 and C4.5 algorithms.

## **2 The ILA Inductive Learning Algorithm**

The ILA inductive learning algorithm that was extended in this study was originally designed for noise-free domains. The induction bias used in ILA is to extract a rule from a set of promising rules for a class if and only if the rule covers the greatest proportion of the number of positive and none of negative examples.

ILA runs in a stepwise forward iteration that cycles at most the number of attributes until all positive examples of a single class is covered. Each iteration searches for a description, or a combination of descriptions, that covers a relatively larger number of training examples of a single class than the other candidates do. Having found such description, ILA generates a rule with the antecedent part consisting of that description. It then marks the examples covered by the rule just generated so that they are not considered in further iteration steps. In other words, the ILA works on a rules-per-class basis. For each class, rules are induced to separate examples in that class from examples in all the remaining classes. This produces an ordered list of rules rather than a decision tree. Description of ILA algorithm is given in Figure 1.

```

For each class attribute value perform
{
 1. Set j, which keeps the size of descriptions, to 1.
 2. While there are unclassified examples in current class
 and j is less than or equal to the number of attributes:
 {
 1. Generate the set of all descriptions in the current
 class for unclassified examples using the current description size.
 2. Update occurrence counts of all descriptions in the current set.
 3. Find description(s) passing the goodness measure.
 4. If there is any descriptions that pass the goodness measure:
 {
 1. Assert rule(s) using the 'good' description(s).
 2. Mark Items covered by the new rule(s) as classified.
 }
 Else increment description size j by 1.
 }
}

```

**Figure 1. ILA inductive learning algorithm**

### 3 Extensions to ILA

Two main problems of ILA can be listed as over-fitting and long learning time. The over-fitting problem is due to the bias that tries to generate a consistent classifier on training data. It is, however, well-known fact that real-world data often includes noisy examples causing over-fitting in the generated classifiers. We developed a novel heuristic function that prevents this bias in the case of noisy examples. We also proposed another modification in order to make ILA faster by considering the possibility of generating more than one rule whenever a set of candidate descriptions pertaining to a class is determined.

The ILA algorithm and its extensions have been implemented by using the source code of C4.5 programs. Therefore, in addition to extensions stated above, the

algorithm has also been enhanced by some features of C4.5, such as rule sifting and default class selection (Quinlan, 1993).

### 3.1 Novel Evaluation Function

In general, an evaluation function's score for a description should increase in proportion to both the number of the positive instances covered, denoted by  $p$ , and the number of negative instances not covered, denoted by  $n_c$ . In order to normalize the score, a simple metric takes into account the total number of positive instances,  $P$ , and negative instances,  $N$ , which is given in (Langley, 1996) as

$$(p + n_c) / (P + N),$$

where the resulting value ranges between 0 (when no positives and all negatives are covered) and 1 (when all positives and no negatives covered). This ratio may be used to measure the overall classification accuracy of a description on the training data. Since as  $P$  and  $N$  are constant this measure can actually be reduced to  $p - n$ , where  $n$  is the number of negatives covered and is equal to  $N - n_c$ .

Now, we may turn to description evaluation metric used in ILA, which can be expressed as follows.

*The description should not occur in any of the negative examples of the current class AND must be the one with maximum occurrence in positive examples of the current class.*

Since this metric assumes, however, no uncertainty to be present in the training data, it searches a given description space to extract a rule set that classifies training data perfectly. It is, however, well-known fact that an application targeting real-world domains should address how to handle uncertainty present in real-world data. Generally, noise tolerant classification requires relaxing the constraint that the induced descriptions must classify the consistent part of training data (Clark & Niblett, 1989), which is equivalent to say that classification methods should generate almost true rules (Matheus, Chan, & Piatetsky-Shapiro, 1993).

The above idea is also supported by one of the guiding principles of soft-computing, "*Exploit the tolerance for imprecision, uncertainty, and particular truth to achieve tractability, robustness, and low solution cost*" (Zadeh, 1994). Using the ideas presented above a new quality criterion is established, in which the quality of a description ( $D$ ) is calculated by the following heuristic:

$$\text{Quality}(D) = p - PF * n$$

Here,

$p$  is the number of correctly identified positive examples,

$n$  is the number of wrongly identified negative examples,

*PF* is a *Penalty Factor*, a user-defined minimum for the proportion of  $p$  to  $n$ .

The *Penalty Factor* is similar to the well-known *sensitivity* measure used for accuracy estimation. Usually *sensitivity* ( $S_n$ ) is defined as the proportion of items that have been correctly predicted to the number of items covered.

$$S_n = p / (p + n)$$

*Sensitivity* may be equivalently rewritten in terms of *PenaltyFactor* ( $PF$ ) as

$$S_n = PF / (PF + 1)$$

The advantage of the new heuristic may be seen by an example case. Let us have two descriptions one with  $p=70$  and  $n=2$ , the other with  $p=5$  and  $n=0$ . The ILA quality criteria select the second description. However, the soft criteria selects the first one, which is intuitively more predictive than the second.

Table 1. Results for splice data-set with different values of penalty factors

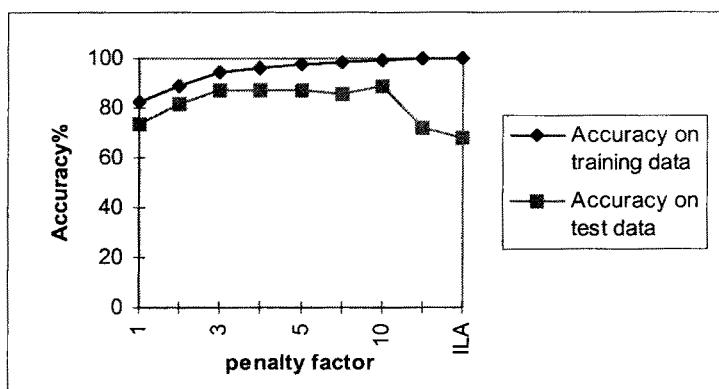
| <i>Penalty Factor</i> | <i>Number of rules</i> | <i>Average number of conditions</i> | <i>Total number of conditions</i> | <i>Accuracy on training data</i> | <i>Accuracy on test data</i> |   |
|-----------------------|------------------------|-------------------------------------|-----------------------------------|----------------------------------|------------------------------|---|
| 1                     | 13                     | 2.0                                 | 26                                | 82.4%                            | 73.4%                        |   |
| 2                     | 31                     | 2.3                                 | 71                                | 88.7%                            | 81.6%                        |   |
| 3                     | 38                     | 2.3                                 | 86                                | 94.7%                            | 87.6%                        |   |
| 4                     | 50                     | 2.5                                 | 125                               | 96.1%                            | 87.2%                        |   |
| 5                     | 53                     | 2.4                                 | 128                               | 97.9%                            | 86.9%                        |   |
| 7                     | 63                     | 2.5                                 | 158                               | 98.7%                            | 85.4%                        |   |
| 10                    | 66                     | 2.5                                 | 167                               | 99.6%                            | 88.5%                        | ✓ |
| 30                    | 87                     | 2.6                                 | 228                               | 100.0%                           | 71.7%                        |   |
| ILA                   | 91                     | 2.6                                 | 240                               | 100.0%                           | 67.9%                        |   |

When penalty factor approaches to the number of training examples the selection with the new formula is the same as ILA criteria. On the other hand, a zero penalty factor means the number of negative training examples has no importance and only the number of positive examples are looked at, which is quite an optimistic choice. Selection of the penalty factor is dependent on domain and the user needs. If the domain data have much noise or/and less processing time is desirable than smaller values are advised.

Properties of classifiers generated by different values of penalty factor for the splice data set are given in Table 1. This data set was chosen as it contains a large number of

attributes and examples within the evaluation sets listed in Table 3. The number of rules and average number of conditions in the rules increase as the penalty factor increases as seen in Table 1. Therefore, ILA with penalty factor constructs always smaller size classifiers for the splice data set.

As seen from Figure 2, for all reasonable values of penalty factors, (1-30), we get better results than ILA in terms of estimated accuracy. For this data set we get maximum estimated accuracy prediction when penalty factor is 10. Increasing the penalty factor further does not give better classifiers. However, when we look at the accuracy on training data we see that there is a linear increase in the accuracy as the penalty factor increases.



**Figure 2. Comparison of accuracy on training and test data for splice data-set.**

### 3.2 Faster Pass Criteria

In each iteration, after an exhaustive search for good descriptions in the search space, ILA selects only one description to generate a new rule. This approach seems an expensive way of rule extraction. However, it is possible that if there exists more than one description with the same quality then all these descriptions may be used for possible rule generation. Usually the second approach tends to decrease the processing time because when all possible descriptions of each trial are asserted more tuples in the training-set are covered than the case of only one description asserted in each trial. On the other hand, this approach might result in redundant rules with an increase in the size of the output rule-set. However, the rule sifting and drop mechanism adapted from C4.5 (Quinlan, 1993) eliminates all those redundant rules.

The above idea was implemented in the ILA system and the option to activate this feature was named as Fast-ILA. For example, in case of promoter data set Fast-ILA reduced the processing time from 17 seconds to 10 seconds. Also, the number of final rules decreased by one and the total number of conditions by two. The experiment

show that, if the size of the classification rules is not extremely important and less processing time is desirable then Fast-ILA option would be more suitable to use.

Table 2. Effect of Fast-ILA option in terms of four different parameters

|                     | <i>Number of initial rules</i> |              | <i>Number of final rules</i> |              | <i>Estimated accuracy (%)</i> |              | <i>Time (second)</i> |              |
|---------------------|--------------------------------|--------------|------------------------------|--------------|-------------------------------|--------------|----------------------|--------------|
| <i>Training set</i> | <i>ILA</i>                     | <i>ILA-2</i> | <i>ILA</i>                   | <i>ILA-2</i> | <i>ILA</i>                    | <i>ILA-2</i> | <i>ILA</i>           | <i>ILA-2</i> |
| <i>Lenses</i>       | 6                              | 9            | 6                            | 5            | 50                            | 62.5         | 1                    | 1            |
| <i>Monk1</i>        | 23                             | 30           | 21                           | 22           | 100                           | 94.4         | 1                    | 1            |
| <i>Monk2</i>        | 61                             | 101          | 48                           | 48           | 78.5                          | 81.3         | 3                    | 2            |
| <i>Monk3</i>        | 23                             | 37           | 23                           | 23           | 88.2                          | 86.3         | 1                    | 1            |
| <i>Mushroom</i>     | 24                             | 55           | 16                           | 15           | 100                           | 100          | 1476                 | 1105         |
| <i>Parity5+5</i>    | 35                             | 78           | 15                           | 14           | 50.8                          | 50.8         | 9                    | 5            |
| <i>Tic-tac-toe</i>  | 26                             | 31           | 26                           | 26           | 98.1                          | 98.1         | 90                   | 52           |
| <i>Vote</i>         | 33                             | 185          | 27                           | 22           | 96.3                          | 94.8         | 31                   | 25           |
| <i>Zoo</i>          | 9                              | 55           | 9                            | 9            | 91.2                          | 85.3         | 1                    | 0            |
| <i>Splice</i>       | 93                             | 825          | 91                           | 76           | 67.9                          | 97.7         | 1569                 | 745          |
| <i>Coding</i>       | 141                            | 1073         | 108                          | 112          | 68.7                          | 100          | 1037                 | 345          |
| <i>Promoter</i>     | 14                             | 226          | 14                           | 13           | 100                           | 100          | 17                   | 10           |

As seen in Table 2, ILA with fast pass criteria generates higher number of initial rules, up to 15 times more rules than ILA. This is because of the faster pass criteria that permit more than one rule to be asserted at once. However, the sifting process, sifts all unnecessary rules.

#### 4 Evaluation of Inductive Learning Algorithm (ILA-2)

For evaluation purposes of ILA-2 we have mainly used two parameters: classifier size and accuracy. Classifier size is the total number of conditions of the rules in the classifier. For decision tree algorithms classifier size refers to the number of leaf nodes in the decision tree; i.e., the number of regions that the data is divided by the tree. Accuracy is the estimated accuracy on test data. We have used the hold-out method to estimate the future prediction accuracy on unseen data.

Table 3. The characteristic features of data-sets

| Domain Name | Number of attributes | Number of examples in training data | Class frequencies in training data                  | Number of examples in test data | Class frequencies in test data                        |
|-------------|----------------------|-------------------------------------|-----------------------------------------------------|---------------------------------|-------------------------------------------------------|
| Lenses      | 4                    | 16                                  | 1-13%, 2-13%, 3-75%                                 | 8                               | 1-25%, 2-37%, 3-37%                                   |
| Monk1       | 6                    | 124                                 | No-50%, Yes-50%                                     | 432                             | No-50%, yes-50%                                       |
| Monk2       | 6                    | 169                                 | 0-62%<br>1-38%                                      | 432                             | 0-67%<br>1-32%                                        |
| Monk3       | 6                    | 122                                 | No-51%<br>Yes-49%                                   | 432                             | No-47%<br>Yes-52%                                     |
| Mushroom    | 22                   | 5416                                | Edible-52%<br>Poisonous-48%                         | 2708                            | Edible-51%<br>Poisonous-48%                           |
| Parity5+5   | 10                   | 100                                 | 0-45%<br>1-55%                                      | 1024                            | 0-50%<br>1-50%                                        |
| Tic-tac-toe | 9                    | 638                                 | Positive-66%<br>Negative-34%                        | 320                             | Positive-64%<br>Negative-36%                          |
| Vote        | 16                   | 300                                 | Democrat-61%<br>Republic-39%                        | 135                             | Democrat-61%<br>Republic-39%                          |
| Zoo         | 16                   | 67                                  | 1-43%, 2-1%,<br>3-1%, 4-3%,<br>5-1%, 6-9%,<br>7-10% | 34                              | 1-35%, 2-18%,<br>3-12%, 4-12%,<br>5-9%, 6-6%,<br>7-9% |
| Splice      | 60                   | 700                                 | EI-30%<br>IE-30%<br>Neither-40%                     | 3170                            | EI-24%<br>IE-24%<br>Neither-52%                       |
| Coding      | 15                   | 600                                 | Coding-50%<br>Noncoding-50%                         | 5000                            | Coding-50%<br>Noncoding-50%                           |
| Promoter    | 57                   | 106                                 | Promoter-50%<br>Other-50%                           | 40                              | Promoter-57%<br>Other-43%                             |

We have used twelve different training sets from the UCI repository (Merz & Murphy, 1997). Table 3 summarizes the characteristics of these data-sets. In order to test the algorithms for the ability of classifying unseen examples a standard practice is to reserve a portion of the training data-set as a separate test set, which is not used in building the classifiers. We have found the test sets related with these data-sets from the UCI Repository, and used them to estimate the accuracy of the classifiers.

We ran three different versions of ILA algorithm on these data-sets: ILA-2 with penalty factor set to 1 and 5, and the basic ILA algorithm. In addition, we ran three different decision tree algorithms and two rule induction algorithms: ID3 (Quinlan, 1986), C4.5 (Quinlan, 1994), OC1 (Murthy Kasif, & Salzberg, 1994), C4.5rules (Quinlan, 1994), and CN2 (Clark & Niblett, 1989). All algorithms were run using the default settings supplied with their systems. Estimated accuracy values of generated classifiers on classifying test sets are given in Table 4 on which our comparative interpretation is based.

Table 4. Estimated accuracy values of the algorithms

| <i>Domain Name</i> | <i>ILA-2 + PF = 1</i> | <i>ILA-2 + PF = 5</i> | <i>ILA</i> | <i>ID3</i> | <i>C4.5-pruned</i> | <i>OC1</i> | <i>C4.5-rules</i> | <i>CN2</i> |
|--------------------|-----------------------|-----------------------|------------|------------|--------------------|------------|-------------------|------------|
| <i>lenses</i>      | 62.5                  | 50                    | 50         | 62.5       | 62.5               | 37.5       | 62.5              | 62.5       |
| <i>monk1</i>       | 100                   | 100                   | 100        | 81.0       | 75.7               | 91.2       | 93.5              | 98.6       |
| <i>monk2</i>       | 59.7                  | 66.7                  | 78.5       | 69.9       | 65.0               | 96.3       | 66.2              | 75.4       |
| <i>monk3</i>       | 100                   | 87.7                  | 88.2       | 91.7       | 97.2               | 94.2       | 96.3              | 90.7       |
| <i>mushroom</i>    | 98.2                  | 100                   | 100        | 100        | 100                | 99.9       | 99.7              | 100        |
| <i>parity5+5</i>   | 50.0                  | 51.1                  | 51.2       | 50.8       | 50.0               | 52.4       | 50.0              | 53         |
| <i>tic-tac-toe</i> | 84.1                  | 98.1                  | 98.1       | 80.9       | 82.2               | 85.6       | 98.1              | 98.4       |
| <i>vote</i>        | 97.0                  | 96.3                  | 94.8       | 94.1       | 97.0               | 96.3       | 95.6              | 95.6       |
| <i>zoo</i>         | 88.2                  | 91.2                  | 91.2       | 97.1       | 85.3               | 73.5       | 85.3              | 82.4       |
| <i>splice</i>      | 73.4                  | 86.9                  | 67.9       | 89.0       | 90.4               | 91.2       | 92.7              | 84.5       |
| <i>coding</i>      | 70.0                  | 70.7                  | 68.7       | 65.7       | 63.2               | 65.9       | 64.0              | 100        |
| <i>promoter</i>    | 97.5                  | 97.5                  | 100        | 100        | 95.0               | 87.5       | 97.5              | 100        |

ILA algorithms have higher accuracy values than C4.5 algorithms in classifying nine out of twelve domains. Similarly ILA algorithms are better than OC1 in seven domains. Compared to CN2, ILA-2 performs better than CN2 in six of the twelve domains and they have similar accuracy in three of them. The best performance of the ILA algorithms is in monk1 domain, in this domain while C4.5 obtained 75.7% accuracy ILA has achieved 100% accuracy. The worst case of ILA, however, is in splice domain, it has minimum 67.9% accuracy for the crisp case and maximum 88.5% accuracy when the penalty factor is 10 (see Table 3), on the other hand OC1 and C4.5 have achieved greater than 90% accuracy in this domain.

Table 5 shows size of the output classifiers generated by the same algorithms above for the same data sets. Results in the table prove that ILA-2 or ILA with the novel evaluation function is comparable to C4.5 algorithms in terms of the generated classifiers' size. When penalty factor set to 1, ILA-2 produces smaller size classifiers than C4.5 algorithms for seven of the twelve domains.

In regard to results in Table 5, it may be worth pointing out that ILA-2 solves overfitting problem of basic (certain) ILA, in a similar fashion to C4.5 solves the overfitting problem of ID3 (Quinlan, 1986). The sizes of classifiers generated by the corresponding classification methods show the existence of these relationships clearly.

Table 5. Size of the classifiers generated by various algorithms

| <i>Domain Name</i> | <i>ILA-2 Penalty Factor 1</i> | <i>ILA-2 Penalty Factor 5</i> | <i>ILA</i> | <i>ID3</i> | <i>C4.5-pruned</i> | <i>C4.5-rules</i> |
|--------------------|-------------------------------|-------------------------------|------------|------------|--------------------|-------------------|
| <i>lenses</i>      | 9                             | 13                            | 13         | 9          | 7                  | 8                 |
| <i>monk1</i>       | 14                            | 37                            | 58         | 92         | 18                 | 23                |
| <i>monk2</i>       | 9                             | 115                           | 188        | 176        | 31                 | 35                |
| <i>monk3</i>       | 5                             | 48                            | 63         | 42         | 12                 | 25                |
| <i>mushroom</i>    | 9                             | 13                            | 22         | 29         | 30                 | 11                |
| <i>parity5+5</i>   | 2                             | 67                            | 81         | 107        | 23                 | 17                |
| <i>tic-tac-toe</i> | 15                            | 88                            | 88         | 304        | 85                 | 66                |
| <i>vote</i>        | 18                            | 35                            | 69         | 67         | 7                  | 8                 |
| <i>zoo</i>         | 11                            | 16                            | 17         | 21         | 19                 | 14                |
| <i>splice</i>      | 26                            | 128                           | 240        | 201        | 81                 | 88                |
| <i>coding</i>      | 64                            | 256                           | 319        | 429        | 109                | 68                |
| <i>promoter</i>    | 7                             | 18                            | 27         | 41         | 25                 | 12                |

## 5 Conclusion

The main contribution of our work is the evaluation metric utilized in ILA-2 for selecting description(s) of a rule. In other words, users can reflect their preferences via a *penalty factor* to tune up (or control) the performance of ILA-2 system with respect to the nature of the domain at hand. This provides a valuable advantage over most of the current inductive learning algorithms.

We also introduced faster pass criteria that reduces the processing time by employing a greedy rule generation bias. This feature, called as *Fast-ILA*, is useful for situations where a reduced processing time is more important than the size of output classifier.

Finally, using a number of machine learning and real-world data sets, we show that the performance of ILA-2 is comparable with that of well-known algorithms, namely CN2, OC1, ID3 and C4.5.

## Acknowledgments

This research is partly supported by the State Planning Organization of the Turkish Republic under the research grant 97-K-12330, and also by the Middle East Technical University under the research grant AFP-97-01-09-01.

All of the data sets were obtained from the University of California-Irvine's repository of machine learning databases and domain theories, managed by Patrick M. Murphy. We acknowledge Ross Quinlan for the implementations of C4.5 and also Rony Kohavi, for we have used MLC++ library to execute OC1, CN2, and ID3 algorithms.

## References

- Clark, P. & Niblett, T., (1989). "The CN2 Induction Algorithm", *Machine Learning*, 3, pp.261-283.
- Deogun, J. S., Raghavan, V. V., Sarkar A., and Sever, H., (1997). "Data Mining: Trends in Research and Development", *Rough Sets and Data Mining: Analysis for imprecise Data*. (T. Y. Lin and N. Cercone, Eds.), Kluwer Academic Publishers.
- Fayyad U.M. (1996). "Data Mining and Knowledge Discovery: Making Sense Out of Data", *IEEE Expert*, October, pp. 20-25.
- Kohavi, R., Sommerfield, D., & Dougherty, J., (1996). "Data Mining Using MLC++: A Machine Learning Library in C++", *Tools with AI*, pp. 234-245.
- Langley, P., (1996). Elements of Machine Learning. San Francisco: Morgan Kaufmann Publishers.
- Matheus, C.J., Chan, P. K., & Piatetsky-Shapiro, G., (1993). "Systems for Knowledge Discovery in Databases", *IEEE Trans. on Knowledge and Data Engineering*, 5(6), pp.903-912.
- Merz, C. J., & Murphy, P. M., (1997). UCI Repository of Machine Learning Databases, <http://www.ics.uci.edu/~mlearn/MLRepository.html>, Irvine, CA: University of California, Department of Information and Computer Science.
- Murthy, S.K., Kasif, S., & Salzberg, S., (1994). "A System for Induction of Oblique Decision Trees", *Journal of Artificial Intelligence Research*, 2, pp.1-32.
- Quinlan, J.R., (1986). "Induction of Decision Trees", *Machine Learning*, 1, pp.81-106.
- Quinlan, J.R., (1993). C4.5: Programs for Machine Learning. Philadelphia, PA: Morgan Kaufmann.
- Quinlan, J.R., (1994). "The Minimum Description Length Principle and Categorical Theories", *Proceedings of the 11th International Conference on Machine Learning*, pp. 233-241.

- Salzberg, S., (1995). "On Comparing Classifiers: A critique of current research and Methods", Technical Report JHU-5/06, Department of Computer Science, John Hopkins University, May 1995.
- Simoudis E. (1996), "Reality Check for Data Mining", *IEEE Expert*, October, pp. 26-33.
- Tolun, M.R., & Abu-Soud, S.M., (1998). "ILA: An Inductive Learning Algorithm for Rule Extraction", to appear in *Expert Systems with Applications*.
- Zadeh, L. A., (1994). "Soft Computing and Fuzzy Logic", *IEEE Software*, pp 48-56.

# Feature Mining and Mapping of Collinear Data

O. de Vel, D. Coomans and S. Patrick

School of Computer Science, Mathematics and Physics,  
James Cook University,  
Townsville 4811, Australia.

**Abstract.** Collinear data such as spectra and time-varying signals are very high-dimensional and are characterized by having highly correlated, context-dependent localized structures. Feature mining involves extracting the important local features whilst, at the same time, retaining as much information as possible and facilitating the automated analysis and interpretation of the data. We present a novel wavelet-based feature mining approach which extracts the optimal features for a particular application. An automated search is performed for the wavelet which optimizes specified multivariate modeling criteria. In this paper we consider mapping analysis as the multivariate model and show how wavelets are able to elucidate the underlying group structure in the data.

**Keywords:** feature mining, collinearity, mapping, adaptive wavelets

## 1 Introduction

Many applications generate collinear data sets which are characterised as extremely multivariate, consisting of many highly correlated features or attributes. Example applications include the recognition of time-varying sonar target signatures, image analysis, and spectroscopy for the non-destructive analysis of chemical substances. Spectral data are obtained by spectrometers which measure the change in reflected or absorbed radiation which has been directed at some substance for hundreds of wavelengths, usually at regular intervals, where each wavelength can be considered equivalent to a feature or attribute. The important characteristics of a spectrum are identified by the localized peak and/or shoulder patterns or global baseline trends. The extraction of such highly collinear features which represent the morphology and/or shifts in spectral peaks are important for interpreting any variations in spectral characteristics which may be due to changes in the chemical composition of a substance or may be indicative

of changes in the underlying structural and molecular parameters (e.g. conformational changes, variations in crystallinity in polymeric fibres etc.).

Processing of such collinear data sets often involves statistical modeling and multivariate data analysis. Depending on the problem at hand, techniques applied include: regression, dimension reduction, discriminant analysis, or cluster analysis [1]. There are several difficulties which arise from analyzing such data sets. One of the major problems is that the dimensionality is quite large (often  $> 1024$  for spectral data), especially when compared to the number of available data instances. Consequently the estimation of parameters becomes highly variable and, in some instances, unobtainable due to numerical instabilities leading to a substantial performance degradation of the multivariate statistical model. A second problem is the existence of a high correlation structure in the data owing to the presence of a strong ordering (collinearity) in the features. The extraction of such highly collinear, context-dependent features are important for modeling and interpreting any variations in the morphological characteristics of the data set.

There exist some statistical and data mining methods which have evolved in recent years with the aim of combating problems associated with the high dimensionality and high correlation feature structures. Such techniques are referred to as high-dimensional techniques and include regularized discriminant analysis (RDA) and penalized discriminant analysis (PDA) in classification or ridge regression in the modeling of data [2]. Conversely, techniques which begin to fail when the dimensionality becomes large when compared to the data sample sizes are referred to as low-dimensional techniques. Low-dimensional techniques include Bayesian linear discriminant analysis, multiple linear regression. Decision trees are also more suited to low-dimensional problems involving fewer attributes. The high-dimensional methods generally allow for a more automated procedure for modeling. Unfortunately though, most high-dimensional methods have evolved quite recently and are therefore not as well publicised or understood by the scientific or industrial community. Finally, most of these techniques provide few facilities for aiding the interpretation of the resulting multivariate

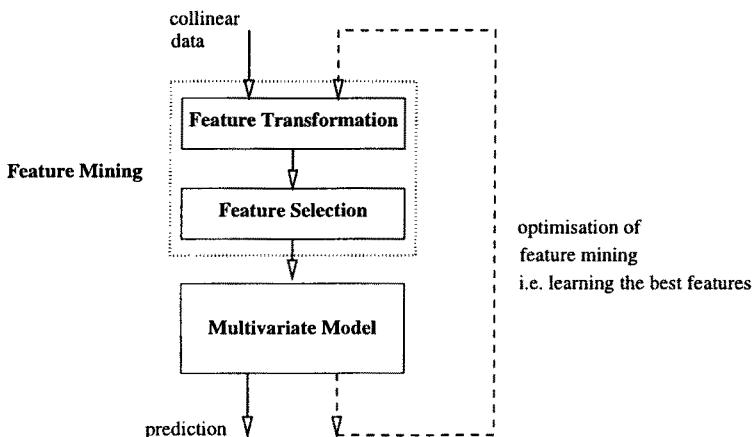
prediction model. For these reasons, low-dimensional methods are often preferred.

Before low-dimensional methods are applied, some form of feature mining should be implemented prior to the interpretation phase. Feature mining is the search for feature or attribute patterns by mapping the data vector  $X$  into a lower-dimensional feature vector  $Y$  (where  $\text{length}(Y) \ll \text{length}(X)$ ). This mapping can be achieved by a linear or non-linear transformation, the transformation chosen so as to retain as much relevant information in the feature patterns as possible, which in itself is dependent on the multivariate data analysis technique advocated. The lower-dimensional feature space may simply be a subset, or a complex transformation, of the original feature space. Having fewer features may result in having features that are more meaningful than the raw features, thereby enhancing the interpretability of the data. Reducing the number of features also means results can often be obtained with a reduced computational expense.

Feature mining can consist of two main components namely, the feature transformation algorithm and the feature selection algorithm. Feature mining can either be independent of, or integrated with, the multivariate prediction model. An *integrated feature mining* procedure makes use of the criterion function(s) of the prediction model (e.g. loss functions) whereas independent feature mining uses criterion functions which are not directly related to the prediction model (see Figure 1). An example independent feature mining procedure is to perform a discriminant analysis on principal components. This procedure is independent because the principle components are obtained by maximizing variability which is independent of the allocation criterion or separability measure (e.g. Wilk's  $\Lambda$  statistic) which are more related (or integrated) with discriminant analysis.

Consideration should also be given to the selection of combinations of collinear features since, for example, two or more collinear features chosen separately will generally produce less favourable results than those features chosen in combination. With high-dimensional data it is not realistic to perform an exhaustive search of every possible combination of features. In this case, forward step-wise selection methods can be considered, whereby features are sequentially incorporated until little or no improvement in the criterion function of the multivariate

model is observed. Another approach is to use a set of localised basis functions, for example wavelets, to represent the context-dependency of the features. A subset of these transformed features can then be used as inputs into the prediction model. Experiments with linear and non-linear discriminant models indicate that step-wise selection methods are rarely superior to methods employing localised basis functions [3].



**Fig. 1. Integrated feature mining.**

The integrated feature mining procedure which we consider, continually updates the features until some modeling criterion becomes optimal. For instance, if we were performing regression analysis on the data we may choose to find the features which optimize a criterion based on the residual sum-of-squares or, a criterion for classification may involve misclassification rates. We propose a novel approach based on a concise and complete parameterisation of the adaptive wavelet matrix which forms the basis of an integrated multivariate feature transformation [4]. The actual features which we continually update are the wavelet coefficients. The wavelet coefficients are produced from a wavelet which is called an adaptive or task-specific wavelet since it is ‘adapting’ to the current data set or task. The adaptiveness is obtained by searching for optimal wavelets trained on a particular collinear data set and using a given prediction model criterion

function that reflects the performance of the model. In this paper we consider the mapping analysis multivariate model to identify the structure and existence of outliers in a data set as a particular application. We describe adaptive wavelets in the next section.

## Wavelet-Based Feature Mining

Many feature transforms have been proposed for collinear data ranging from univariate to multivariate transformations involving all the features of the spectrum. A multivariate technique commonly used is principal components analysis (PCA). PCA transforms the original features into a new set of uncorrelated features that are linear combinations of the original variables derived in decreasing order of variability. That is, PCA identifies the direction of greatest variance corresponding to the largest eigenvector. Of particular importance with collinear data is the ordering of the features. Although PCA is a valid data mapping technique (see the next section) it, unfortunately, does not take into account the ordering, and hence the correlational structure of the features. Alternatively, the Fourier transform (FT) can be used to take into account the ordering of variables in a spectrum. However, the FT is a global transform and any localised changes in a data instance will affect most, if not all, of the Fourier coefficients. To avoid such global effects and to better identify local feature patterns, the wavelet decomposition can be used for variable transformation. The wavelet decomposition is a basis function expansion into localised contributions labeled by scale and a position (feature index) parameter [5]. The coefficients in the expansion of the wavelet basis functions (called wavelet coefficients) are able to convey small-scale effects in correlated feature patterns as well as larger scale fluctuations such as baseline changes. Also, one useful property of the wavelet transform is its ability to backtransform the wavelet coefficients, thereby enabling an improved interpretation of the relevant (extracted) features.

We can introduce wavelets in different ways. For example, through multiresolution analysis (MRA), we can construct a variety of wavelets in a unified setting [6]. MRA is convenient for deriving the lattice decomposition and reconstruction equations of the wavelet transform for infinite signals. In general,

however, we do not have access to infinite discrete signal or spectra and we are obliged to represent an infinite sequence by the periodic extension of the finite (compact) signal. This gives rise to the discrete wavelet transform (DWT).

### The Discrete Wavelet Transform (DWT) and Adaptive Wavelets (AW)

Since the wavelet coefficients are the features which are eventually supplied to the multivariate procedures, we now focus our attention on how the wavelet coefficients are calculated. The procedure for calculating the wavelet coefficients is called the discrete wavelet transform. For further, more detailed, background information on wavelets see, for example, [7][8].

We can write the decomposition/reconstruction equations of the DWT as [6],

$$\mathbf{f}_{j-1} = C_N(\mathbf{A}) \mathbf{f}_j^{(0)} \quad \text{and} \quad \mathbf{f}_j^{(0)} = C_N(\mathbf{A})^T \mathbf{f}_{j-1}$$

where  $f_{j,k}^{(0)} = \langle f, m^{j/2} \phi(m^j x - k) \rangle \equiv \langle f, \phi_{j,k}(x) \rangle$  for some *scaling* function  $\phi$ , dilation parameter  $m$  (i.e. the number of bands in the DWT), translation  $k$  and,  $f_{j,k}^{(s)} = \langle f, m^{j/2} \psi^{(s)}(m^j x - k) \rangle \equiv \langle f, \psi_{j,k}^{(s)}(x) \rangle$  for  $s = 1, 2, \dots, m-1$  for some *wavelet* functions  $\psi^{(s)}$ , a linear combination of the scaling function. The index  $j$  indicates the level in the  $m$ -way tree DWT decomposition scheme.

The compact vectors  $\mathbf{f}_j^{(0)} = (f_{j,0}^{(0)} \ f_{j,1}^{(0)} \ \dots \ f_{j,mN-1}^{(0)})^T$  and

$$\mathbf{f}_{j-1} = (f_{j-1,0}^{(0)} \ f_{j-1,1}^{(1)} \ \dots \ f_{j-1,m-1}^{(m-1)} \ f_{j-1,1}^{(0)} \ f_{j-1,2}^{(1)} \ \dots \ f_{j-1,N-1}^{(m-1)})^T$$

are  $mN$ -periodic.  $C_N(\mathbf{A})$  is a block circulant matrix generated by the *wavelet matrix*  $\mathbf{A}$  (where  $q = N/m$ ) and has the special “banded” structure:

$$C_N(\mathbf{A}) = \begin{pmatrix} \mathbf{A}_0 & \mathbf{A}_1 & \dots & \mathbf{A}_{q-2} & \mathbf{A}_{q-1} & 0 \\ 0 & \mathbf{A}_0 & \dots & \mathbf{A}_{q-3} & \mathbf{A}_{q-2} & \mathbf{A}_{q-1} \\ \vdots & \vdots & \dots & \vdots & \vdots & \vdots \\ \mathbf{A}_1 & \mathbf{A}_2 & \dots & \mathbf{A}_{q-1} & 0 & \mathbf{A}_0 \end{pmatrix}$$

where  $\mathbf{A}$  is composed of  $q$  sub-matrices  $\mathbf{A}_i$  (for  $i = 0, 1, \dots, q-1$ ) each of size  $m \times m$  such that  $\mathbf{A} = (\mathbf{A}_0 \ \mathbf{A}_1 \ \mathbf{A}_2 \ \dots \ \mathbf{A}_{q-1})$ . The wavelet matrix  $\mathbf{A}$  denotes the matrix of filter coefficients, with the first row containing the low-pass coefficients and the remaining  $m-1$  rows, the sets of high-pass coefficients. Certain regularity

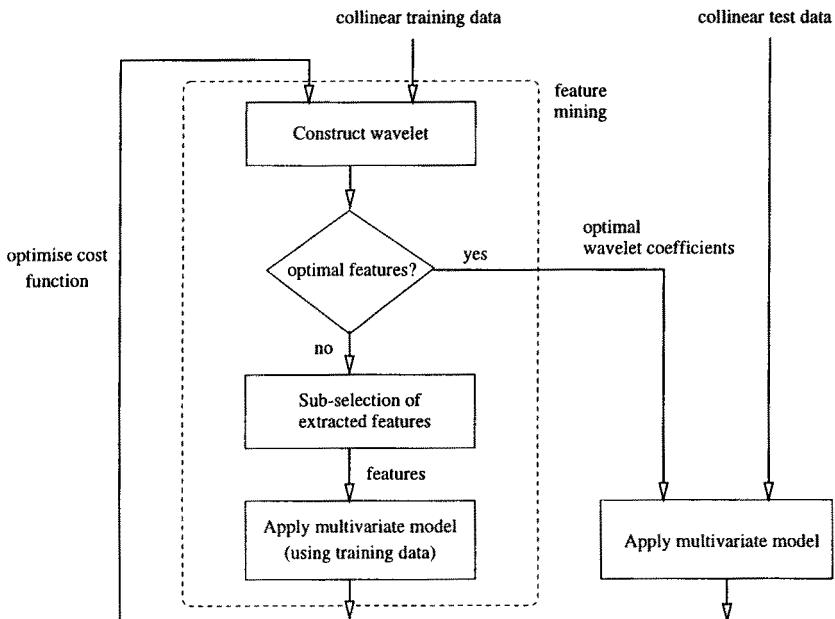
conditions on the wavelet matrix  $\mathbf{A}$  must be satisfied if  $C_N(\mathbf{A})$  is to be a banded orthogonal block circulant matrix such as, for example, shifted orthogonality.

In practice, “off-the-shelf” values for the low- and high-pass filter coefficients can be chosen from the literature. These are readily available more so for the situation when  $m = 2$ , for example the Daubechies wavelet family. We believe, that it can be advantageous to design your own task specific filter coefficients rather than using a predefined set. The problem we pose is to be able to describe a large class of orthogonal compactly-supported wavelets with a given length of support and design the wavelet matrix  $\mathbf{A}$  which optimizes some specified modeling criterion relevant to a given multivariate prediction model, such as regression, discriminant or mapping analysis. Instead of optimizing over each element in  $\mathbf{A}$ , we make use of the factorized form [4] of a wavelet matrix and the regularity conditions placed therein to reduce the number of parameters to be optimized. It can be shown that  $\mathbf{A}$  can be constructed from some set of normalized vectors, denoted by  $\mathbf{u}_1, \dots, \mathbf{u}_q$  and  $\mathbf{v}$ . We can update these vectors by using a search procedure based on minimizing some application-dependent cost/criterion function. Initial values for these vectors are generally chosen to be random values. Example cost functions for discriminant analysis include misclassification rate, entropy, Wilks’  $\Lambda$  statistic and the quadratic probability measure. The construction of these matrices and algorithms for integrated feature mining using adaptive wavelets (AW) are described in more detail in [3].

### **Subset Selection of the Wavelet Coefficients**

Since there are just as many wavelet coefficients as there are wavelengths in the original data set, it is necessary to perform a compression or a subset selection on the wavelet coefficients. This can be achieved by, for example, using a best-basis algorithm which minimizes some information cost function or by selecting a set of wavelet coefficients from a predetermined band that gives the lowest cost value for the multivariate task at hand [9][3]. In our implementation we choose to select a single band of wavelet coefficients at each level in the DWT. These coefficients are then supplied to the multivariate procedure. The modeling criterion for optimizing the wavelet matrix is also based on the same coefficients.

Figure 2 shows a schematic diagram of the learning process involved in extracting the optimal features for a given cost criterion and multivariate model. Note that, in the case of implementing cross-validation for small data sets, an additional external loop would need to be incorporated.



**Fig. 2.** Learning the optimal features using adaptive wavelets.

## Data Mapping Analysis

Data mapping is a multivariate data analysis technique often used to i) reduce the dimensionality of the data set, ii) analyse the structure of the data set or, iii) identify the presence or absence of outliers. Different linear and non-linear data mapping techniques are available. A well-known linear technique is principal component analysis (PCA); non-linear techniques include Sammon mapping and multidimensional scaling (MDS). Linear techniques are often preferred as analytical solutions are available and the interpretation of the reduced feature space is more straightforward than for the case of non-linear techniques.

We have investigated the amount of heterogeneity (preservation of topology) contained in a data set with data mapping using an integrated feature mining approach with adaptive wavelets (AW) and the PCA multivariate model. The AWs were used for extracting the relevant collinear features and PCA as the multivariate data analysis technique for reducing the dimensionality of the data set and evaluating the degree of heterogeneity in the data. Different cost functions were used in the PCA multivariate model to optimise the choice of wavelet coefficients namely, a) maximisation of the percentage of variance contained in the first principal component (COST A) b) maximisation of the total percentage of variance contained in the first two principal components (COST B) and, c) maximisation of the product of the variances of the first two principal components (COST C). Cost function COST C causes the simultaneous maximisation of the variances of the first two principal components.

The different parameters used in the AW algorithm (for example, number of bands  $m$ , factorisation product value  $q$ , and tree decomposition value  $l$ ) are determined by some preliminary experiments with the AW algorithm and PCA model and examination of the resulting PC scatterplots.

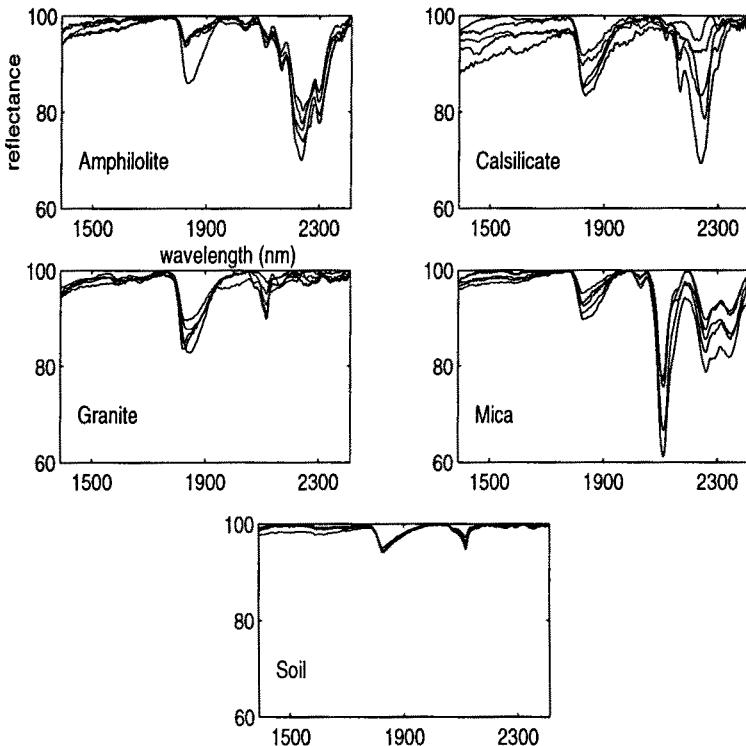
Evaluation of the degree of heterogeneity or group separation contained in the data set after application of PCA can be undertaken in one or more qualitative or quantitative ways, for example; the stress statistic, pairwise distance plots, PC scores scatterplots, Wilk's  $\Lambda$  statistic, generalised procrustes analysis and the permutation test. We have used the Wilk  $\Lambda$  statistic as a simple quantitative measure of group separation. This statistic calculates the ratio of the within-to-between covariance for all groups. A small value for Wilk's  $\Lambda$  indicates good group separation.

## **The Data Sets**

In our experiments we have used spectral data sets as example collinear data. Two spectral data sets, Mineral and Seagrass, were used for evaluating the performance of the AW+PCA procedure (see Table 1). Both data sets have 512 features (wavelengths) and known groups/classes. Some sample spectra from the different classes of the mineralogical data set is shown in Figure 3.

| Data Set | Class 1 | Class 2 | Class 3 | Class 4 | Class 5 | Total |
|----------|---------|---------|---------|---------|---------|-------|
| Mineral  | 20      | 20      | 20      | 20      | 20      | 100   |
| Seagrass | 55      | 55      | 55      | N/A     | N/A     | 165   |

**Table 1.** Description of the spectral data sets used in the experiments.



**Fig. 3.** Sample spectra from the Mineral data set (in 5 mineral classes).

## Results and Discussion

The results of applying principle component analysis on the raw Mineral and Seagrass data sets are shown in Table 2.

The PC scores scatterplot for the Mineral data set is shown in Figure 4. The scatterplot reveals a V-shaped struture in the data, with the spectra lying along two distinct arms. There are visible groupings in the data, although some

overlap exists between spectra of different classes/groups. Further analysis using pairwise Euclidean distance plots indicates that large pairwise distances in the original space tend to be preserved by the PCA mapping whereas the shorter distances are generally not preserved.

| Data Set | PC1 Variance | PC2 Variance | Total  | Wilk $\Lambda$ Statistic |
|----------|--------------|--------------|--------|--------------------------|
| Mineral  | 55.92%       | 23.30%       | 79.31% | 0.033                    |
| Seagrass | 76.62%       | 13.29%       | 89.91% | 0.406                    |

Table 2. Results from PCA on raw spectral data sets.

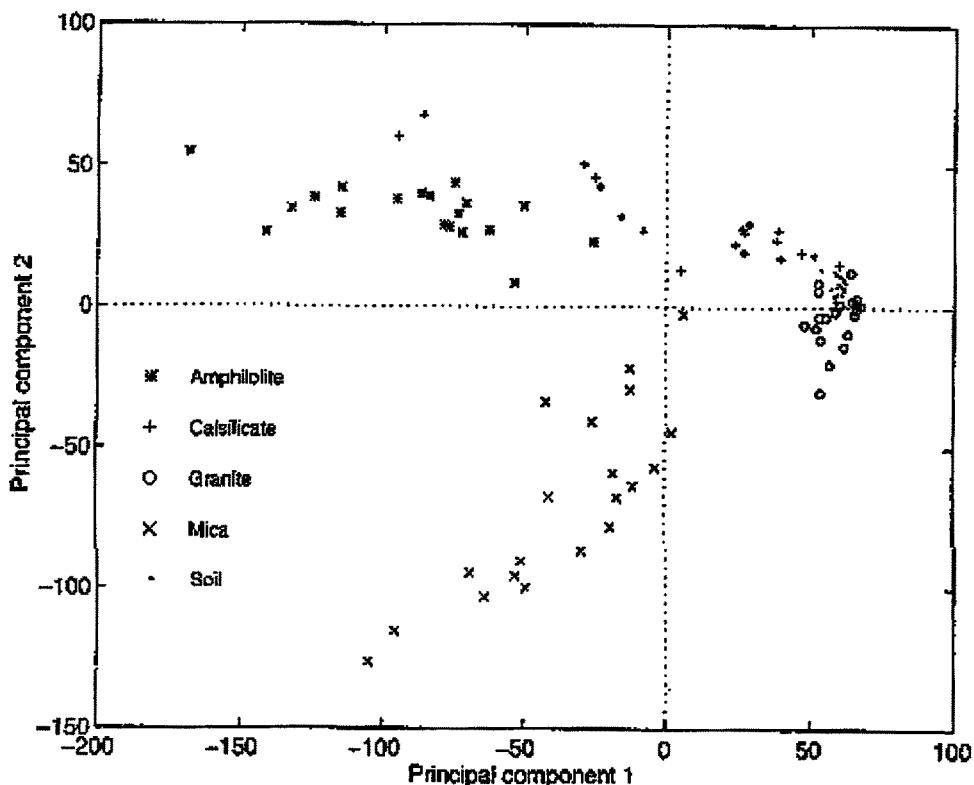


Fig. 4. PC scores scatterplot after PCA on the raw Mineral spectral data set.

The PC scores scatterplot for the Seagrass data set provides some evidence of clustering but with extensive overlap between classes. The pairwise distance

plots show better preservation of shorter distances than in the case of the Mineral data set.

Results for the integrated AW feature extraction + PCA multivariate procedure are shown in Tables 3 and 4 for the case of the Mineral and Seagrass data sets, respectively. The optimal parameter values ( $m, q, l$ ) were obtained by searching over different parameter values and measuring the minimal Wilk  $\Lambda$  statistic. These results show an improvement in the total variance for the first two principle components (and, in particular, for the first principle component with the COST A cost function) as well as a significant reduction in the Wilk's  $\Lambda$  statistic. This indicates that the adaptive wavelets are better able to maximise the variance in the PCs. Also, the number of wavelet coefficients was 32, corresponding to a significant reduction in the number of features.

| Cost Function | PC1 Variance | PC2 Variance | Total | Wilk $\Lambda$ Statistic |
|---------------|--------------|--------------|-------|--------------------------|
| COST A        | 68.19        | 12.33        | 80.52 | 0.026                    |
| COST B        | 55.62        | 27.93        | 83.55 | 0.028                    |
| COST C        | 50.66        | 31.29        | 81.75 | 0.022                    |

Table 3. Results from PCA after applying the AW algorithm on the Mineral data set (for  $(m, q, l) = (4, 2, 2)$ ).

| Cost Function | PC1 Variance | PC2 Variance | Total | Wilk $\Lambda$ Statistic |
|---------------|--------------|--------------|-------|--------------------------|
| COST A        | 99.83        | 0.09         | 99.92 | 0.243                    |
| COST B        | 99.81        | 0.13         | 99.94 | 0.198                    |
| COST C        | 51.82        | 28.11        | 79.93 | 0.277                    |

Table 4. Results from PCA after applying the AW algorithm on the Seagrass data set (for  $(m, q, l) = (4, 2, 2)$ ).

PC scores scatterplots for the AW+PCA procedure show a significant improvement in the groupings in both data sets. For example, Figure 5 for the Mineral data set shows how the "soil" mineral group is now clearly separate from the "granite" and "calsilicate" groups. Furthermore, the directions of the V-structure now appear to be orthogonal suggesting the existence of independent "latent" variables. A similar, more pronounced behaviour is observed for the Seagrass data set. Also, pairwise Euclidean distance plots (after applying

the AW+PCA procedure) enhance the short pairwise distances in the data compared with no wavelets [10]. That is, wavelets focus on specific features of the spectra to enhance the groupings that are present and in doing so preserve the short pairwise distances.

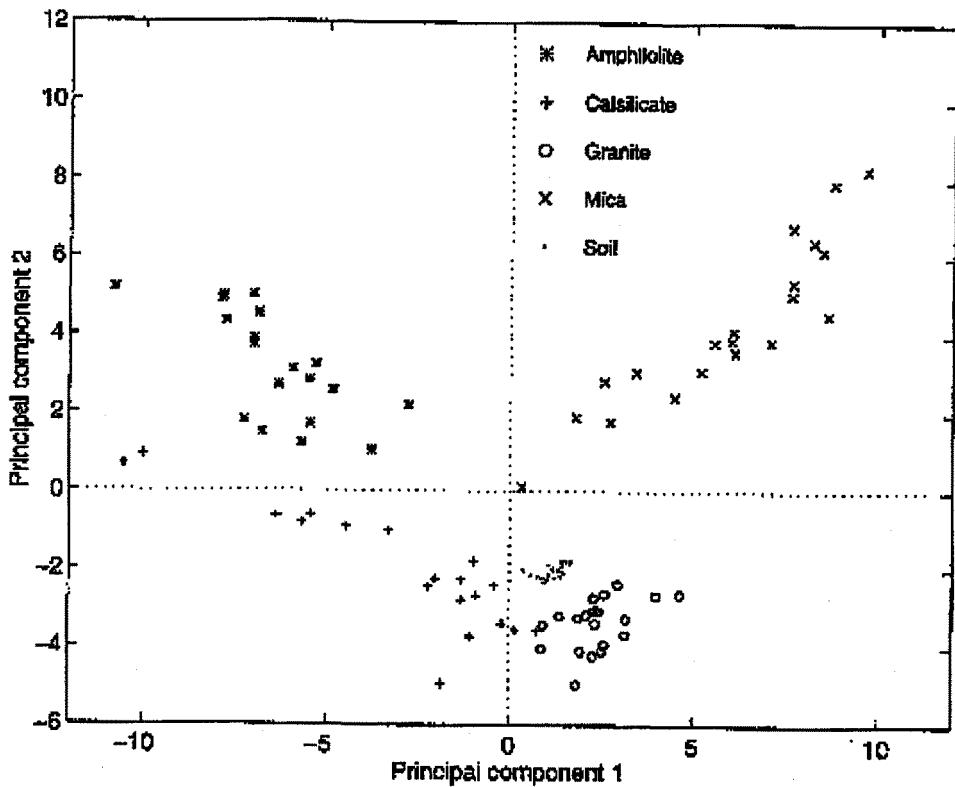


Fig. 5. PC scatterplot after AW+PCA on Mineral spectral data set for cost function COST B.

As was mentioned previously, wavelets offer the possibility of performing a back-transformation to facilitate the identification of the salient features mined from the spectrum. Indeed, it was observed experimentally that the wavelets are able to pick out the regions of the spectra which are characterised by the greatest variability between the different spectral groups.

## Conclusions

Feature mining with wavelets in the context of the multivariate PCA model has resulted in a number of observations: i) a greater proportion of the data variance is contained in the reduced space, ii) an improvement in the separation of the groups/clusters, iii) the possible existence of independent latent variables and, iv) a significant reduction in the number of features together with an identification of the salient features in the data set. Feature mining of collinear data with wavelets has shown to produce results that are consistently equal to, or better than, other multivariate approaches [11][3].

Future extensions will involve more detailed simulations (more data sets), compare results with non-linear mapping techniques and cluster analysis. Some investigations are also under way to study the impact of the wavelet decomposition scheme (e.g. depth and number of bands) on performance, and better heuristics for the subset selection of wavelet coefficients.

## References

1. Fukunaga K.: *Introduction to Statistical Pattern Recognition*. Academic Press, New York (1997)
2. McLachlan G.: *Discriminant Analysis and Statistical Pattern Recognition*. John Wiley (1992)
3. Mallet Y., Coomans D., de Vel O.: Classification using Adaptive Wavelets for Feature Extraction, *IEEE Trans. on Pattern Analysis and Machine Intelligence* **19** (1997) 1058–1066
4. Turcajova R. and Kautsky J.: Shift Products and Factorizations of Wavelet Matrices, *Numerical Algorithms* **8** (1994) 27–45
5. Daubechies I.: *Ten Lectures on Wavelets*, Science for Industrial and Applied Mathematics (1992)
6. Mallat S.: A Theory for Multiresolution Signal and Decomposition: The Wavelet Representation, *IEEE Trans. on Pattern Analysis and Machine Intelligence* **11** (1989) 674–693
7. Chui C.: *An Introduction to Wavelets*. Academic Press, San Diego (1992)
8. Cohen A., Kovacevic J.: Wavelets: The Mathematical Background, *Proceedings of the IEEE* **4** (1989) 514–522
9. Saito N., Coifman R.: Local Discriminant Bases, *Mathematical Imaging: Wavelet Applications in Signal and Image Processing SPIE* **2303** (1994)
10. Patrick S.: Mapping Spectral Data using Adaptive Wavelets. Dept of Mathematics and Statistics, James Cook University, Australia (1996)
11. Mallet Y., Coomans D., de Vel O.: Recent Developments in Discriminant Analysis on High Dimensional Spectral Data, *Journal of Chemometrics and Intelligent Laboratory Systems* **35** (1996) 157–173

# **Knowledge Discovery in Discretionary Legal Domains**

**John Zeleznikow**

Database Research Laboratory, Applied Computing Research Institute,

La Trobe University, Bundoora, Victoria, Australia, 3083

E-mail: [johnz@latcs1.cs.latrobe.edu.au](mailto:johnz@latcs1.cs.latrobe.edu.au); Phone: (61) 3 9479 1003; Fax: (61) 3 9479 3060

**Andrew Stranieri**

School of Information Technology and Mathematical Sciences,

University of Ballarat, Ballarat, Victoria, Australia, 3353

E-mail: [a.stranieri@ballarat.edu.au](mailto:a.stranieri@ballarat.edu.au); Phone (61) 3 53279440 ; Fax (61) 3 53279270

## **Abstract**

Significant obstacles must be overcome if knowledge discovery techniques are to be applied in the legal domain. In this paper we argue that in order to use knowledge discovery in the legal domain it is essential to use domain expertise and important that an abundance of commonplace cases is available.

Even with appropriate data, data mining techniques in law must deal with contradictory cases and use statistical techniques in order to define error and estimate performance. We illustrate these points by describing our own error heuristic and the method we use for dealing with contradictions for the training of neural networks in the domain of property proceedings in Australian Family Law.

In law, an explanation for a decision reached is often more important than the decision. We advocate the use of a theory of argumentation developed by the British philosopher Stephen Toulmin to provide explanations to support the outcomes predicted by our knowledge discovery system Split Up. We also discuss the use genetic algorithms to minimise the number of features our knowledge discovery system must use.

**Keywords:** Knowledge Discovery in Legal Domains, Neural Networks, Explanation, Feature Selection

## **1. Introduction**

[Fayyad et al. 1996a] define knowledge discovery in databases as the non trivial process of identifying valid, novel, potentially useful understandable patterns in data. Data mining is a problem-solving methodology that finds a logical or mathematical description, eventually of a complex nature, of patterns and regularities in a set of data. [Decker and Focardi 1995] state that in practical applications, data mining is based on two assumptions:

1. The functions that one wants to generalise can be approximated through some relatively simple computational model with a certain level of precision;
2. The sample data set contains enough information to perform the generalisation.

The second assumption is a major obstacle to the use of knowledge discovery techniques in the legal domain in jurisdictions based on civil law and also in those based on common law. In common law countries, lawyers reason with a specific type of case, a precedent. A precedent is an adjudged case or decision of a court, considered as furnishing an example or authority for an identical or similar case afterwards arising for a similar question of law. In such countries, courts attempt to decide new cases on the basis of principles established in prior cases [Lyons 1985].

### **1.1 Using Cases to Build Legal Knowledge Based Systems**

The earliest form of retrieving and reasoning with legal cases involved simulating case based reasoning by using logic or production rule systems. Examples included the TAXMAN project [McCarty 1977] and the work of [Gardner 1987]. This concurs with the statement of [Fayyad *et al* 1996c] that: *the traditional method of turning data into knowledge relies on manual analysis and interpretation.*

Despite the fact that artificial intelligence and law researchers have not focussed upon knowledge discovery in database techniques we believe such techniques can be fruitfully applied to analyse legal domains. [Rissland and Friedmann 1995] used rule induction to analyse a domain in order to detect a change in the way a legal concept is used by Courts. Whilst it is accepted that legal concepts change over time, developing computer techniques to detect such change is non-trivial. [Rissland and Friedmann 1995] induced decision trees that represented US bankruptcy law cases at various time intervals over a ten year period.

[Pannu 1995] used knowledge discovery techniques to identify a prototypical exemplar of cases within a domain. An exemplar pro-defendant case has features that are most like the cases in which the defendant won and most unlike the cases the defendant lost. [Merkl and Schweighofer 1997] performed a case study of legal document classification. The core task of classification is performed by a non-standard neural network model with a layered architecture consisting of mutually independent unsupervised neural networks. The resulting system has a remarkably fast training time and explicit cluster representation. [Wilkins and Pillaipakkamatt 1997] examine the feasibility of using machine learning techniques for the task of predicting the elapsed time between the arrest of an offender and the final disposition of her/his case.

### **1.2 Knowledge Discovery in Discretionary Domains**

The knowledge discovery exercises outlined above had focussed on domains of law that were not discretionary. [Black 1990] views discretion as a power or right conferred upon decision-makers to act according to the dictates of their own

judgement and conscience, uncontrolled by the judgement or conscience of others. Nevertheless, decision-makers must act in accordance with the rule of law and their decisions must preserve rights of all parties effected by the decision-making. It is thus essential that discretionary decision-making not be arbitrary — since an arbitrary application of discretion could lead to enhanced conflict by any aggrieved parties.

Few legal reasoning systems have been developed in discretionary domains. [Edwards and Huntley 1992] applied rule-based reasoning to the discretionary domain of Family Law in Scotland and reported some inadequacies of that approach. In [Zeleznikow and Stranieri 1995] we reported on the use of neural networks for the prediction of Court decisions. In that study, we collected data from commonplace cases dealing with property distribution in Australian Family Law. Our aim was to predict what percentage of the marital property an Australian Family Court judge would award to each partner of a failed marriage. A full description of the resultant system, Split Up, can be found in [Stranieri *et al* 1998].

Our experience with the Split Up project led us to conclude that for knowledge discovery in database techniques(KDD) to be applied usefully in legal domains, decisions taken and assumptions made in the selection, pre-processing, transformation and data mining phases of the KDD must be made explicit. In this paper we suggest that commonplace cases are far more appropriate for the selection phase than landmark cases. We also illustrate a metric we used to detect outliers in family law data in the pre-processing stage. The transformation phase was performed in Split Up with the use of a hierarchy of attributes, that represents a tree of arguments based on Toulmin's theory of argumentation [Toulmin 1958]. The data mining phase was performed with neural networks — the performance of each was verified using cross validation resampling.

## 2. Landmark and Commonplace Cases

### 2.1 Landmark Cases

[Fayyad *et al* 1996b] claim that data mining functions can be categorised thus: summarisation, classification, regression and clustering. A number of case oriented artificial intelligence paradigms rely on classification. Examples include neural networks, rule induction and statistically oriented case based reasoners. In doing so, these paradigms assume that the experience being modelled has the same context. If the contexts of the cases differ then the expressions used, the factors relevant, and the outcomes of the cases will necessarily mean different things.

[Kolodner 1993] incorporates context in her definition of a case for case based reasoning systems. She states that 'a case is a contextualised piece of knowledge representing an experience that teaches a lesson fundamental to achieving the goals of the reasoner'. [Zeleznikow *et al* 1997] note that even in non-contentious areas, Kolodner's definition provides scope for considerable problems. They disagree with Kolodner that a case necessarily 'teaches a lesson fundamental to the reasoner'.

Certainly some cases do fit this description. Most notably within law, those decisions from appellate courts which form the basis of later decisions and provide guidance to lower courts do provide a fundamental lesson, or normative structure for subsequent reasoning. The common name for such cases are landmark cases.

However, most decisions in any jurisdiction are not landmark cases. Most decisions are commonplace, and deal with relatively minor matters such as vehicle accidents, small civil actions, petty crime, divorce, and the like. These cases are rarely, if ever, reported upon by court reporting services, nor are they often made the subject of learned comment or analysis. Landmark cases, individually have a profound effect on the subsequent disposition of all cases in that domain, whereas commonplace cases will only have a cumulative effect, and that effect will only be apparent over time.

Take, for example, the case of *Mabo v Queensland (No.2)* [(1992) 175 CLR 1]. Prior to *Mabo* the indigenous people of Australia, the aborigines, had few if any proprietary rights in Australian land. In *Mabo* the High Court held that previous decisions holding that Australia was *terra nullius* (empty land) at settlement, and decisions holding that Aborigines had no property laws affecting land, were simply wrong at law. Hence, the High Court said, Aborigines had sovereignty over parts of Australia under certain conditions. *Mabo* is the landmark case in the area, and will form the basis of future decisions in this area. Indeed, this case like many other leading cases, was the spur for political action and we soon saw the introduction of the Federal *Native Title Act*. Thus, landmark cases have the dual effect of determining (to some degree) the interpretation of subsequent fact situations as well as influencing the invocation of normative legislative processes.

## 2.2 Commonplace Cases and Knowledge Discovery

[Zeleznikow *et al.* 1997] define a commonplace case as one which does not provide any lessons by itself, but together with numerous like cases can be used to derive conclusions. In [Zeleznikow and Stranieri 1997a] they refine the definition to state that a commonplace legal case is one that is not reported in a court reporting service — and hence is not known to any legal practitioners not associated with the given case.

Most legal reasoning systems augment a representation of landmark cases with heuristics used by practitioners of law. Although experts learn these heuristics as a result of constant exposure to commonplace cases there have been few attempts to induce the knowledge directly from case judgments. This is so for a number of reasons:

- commonplace cases judgments are written reports therefore data about each judgment needs to be extracted manually. This is a laborious task prone to error.
- data mining techniques have only recently been developed. Empirically based legal research discovers new legal knowledge by using data sets to

test hypotheses. Until recently there was no possibility of using data sets to discover associations that researchers had not contemplated.

We argue that data mining techniques can be useful for gleaning knowledge in legal domains in which an abundance of data exists in the form of commonplace cases. Rule induction, typified by the ID3 algorithm of [Quinlan 1986], uses information theory to classify cases into sets based on factor-attributes, and then is able to derive rules from this classification. The addition of a single new case to an induction system, should not drastically alter the knowledge of the system. With landmark cases, simple classification along existing lines is inadequate. What these types of cases do is completely recreate the necessary and sufficient conditions for each outcome. This means that, in essence, new factors must be created for the entire training set. Thus, unless the creator of the induction system is aware of the relative importance of each case, the output derived by the induction algorithm will be flawed. Hence, the cases used by rule induction systems should be the commonplace ones, which have a cumulative effect on the law and which reflect the law, rather than which completely re-create the law in a new image. This means that landmark cases must be avoided in the construction of induction systems.

In the next section we shall discuss how we retrieved commonplace cases from the records of the Family Court of Australia and how we used these commonplace cases to perform data mining through the Split Up system.

### **3. Knowledge Discovery in the Split Up system**

#### **3.1 Connectionism and Data Mining**

In contrast to rule-based reasoning, connectionism is suited to modelling discretionary reasoning [Zeleznikow and Stranieri 1997b]. Domain knowledge of legal rules and principles is not modelled directly; nor are a select number of landmark cases retrieved and adapted as occurs when using case-based reasoning. Instead, a connectionist learning algorithm is exposed to data from a large number of cases previously decided so that the way judges have actually exercised discretion in weighing relevant factors can be assimilated into the program.

Neural networks have rarely been used in the legal domain because explanations are difficult to generate and assembling training sets of sufficient size and coverage is similarly difficult. Our approach has been that connectionism can be useful in law if a series of smaller, interconnected networks are used instead of one larger network and if explanations are generated independently of the process used to infer a conclusion. To provide explanation independently of the conclusion inferred we used Toulmin Argument Structures.

[Toulmin 1958] concluded that all arguments consist of four invariants: claim, data, warrant and backing. The assertion of an argument stands as the claim of the argument. Knowing the data and the claim does not necessarily convince one that the claim follows from the data. A mechanism is required to justify the claim given the

data. This justification is known as the warrant. The backing of an argument supports the validity of the warrant. In the legal domain it is typically a reference to a statute or a precedent.

Ninety four factors that were relevant in determining a percentage split in the Split Up system were elicited from experts (principal expert was Renata Alexander from the Legal Aid Commission of Victoria) and from statutes to form a hierarchy of relevant factors. Figure 1 depicts three, of the thirty five arguments used in Split Up. The three arguments labelled A, B and C.

The culminating (right-most) argument, A has, as its claim, the percentage of assets awarded to the husband. This is inferred (using a feed forward neural network represented as a dotted arc) from the contributions of the husband relative to the wife, the future needs of the husband relative to the wife and the wealth of the marriage. These three factors are the data items of the argument. The warrant and backing components of the Toulmin structure are omitted from Figure 1 but are accessed by the user in search of an explanation. Thus, if the user prompts the system to explain why the wife is likely to receive 60% of the assets, she is told that "*although contributions have been equal, the wife has greater future needs in a marriage of average resources.*" This reflects the data components of an argument. Prompting for further explanation results in the retrieval of warrant and backing components.

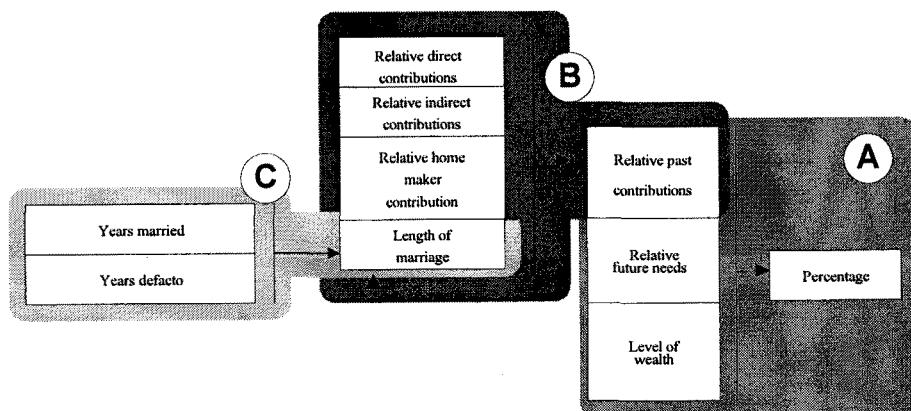
*"Contributions to the marriage are relevant by Section 72(4) of the Family Law Act, Future needs must be taken into account by virtue of Section 75(2) and case law Lee v Lee indicate that the level of resources of the marriage can impact on the contributions / needs balance".*

We see from Figure 1 that the claim of Argument B is used as a data item for Argument A. The claim of argument B is inferred from its data items with the use of a neural network. The claim of argument C has been inferred from its data items with the use of rule sets and is itself another data item for B. A total of thirty-five arguments are used in Split Up. Twenty of the Split Up argument structures have a feed forward neural network trained with backpropagation of errors as the inference procedure. The remaining argument structures make use of small rule sets.

The hierarchy of arguments is important for the generation of user directed explanations. If a user is not convinced that the husband and wife contributed equally in the past (data item for argument A) then the argument that inferred that outcome is retrieved (B) and its data items are displayed: "*The husband contributed directly by salary earnings to a far greater extent than the wife, however the wife contributed as homemaker far more than the husband. This was a long marriage and indirect contributions are equal.*"

The hierarchy of arguments is critical in decomposing the task into smaller sub tasks in the data transformation phase so that data mining techniques can be applied more effectively to each sub task. In [Skabar et al 1997] we use the 94 attributes in Split Up data in a flat structure without segregating the attributes into independent arguments and found that the best data mining methods we used were only able to predict 35% of cases outcomes.

Data for the Split Up system was initially extracted from four hundred written unreported cases. However many of these were considered unsuitable for the task of data mining the training set. For example we eliminated all cases dealing with arguments about the custody of children, since litigants often appear to fight about the custody of children when their real aim is to gain a greater share of property. Eventually we used one hundred and three unreported cases where the only issue of conflict was property — these cases included marriages with children, but in such cases the welfare of the children was not in dispute.



**Fig. 1. Data and claim components of three arguments from Split Up**

The decision as to whether we should use rule sets or neural networks as the inference procedure for an argument depended on a scheme for classifying legal tasks for data mining suitability [Stranieri *et al* 1998]. The classification scheme which helped us determine which family law tasks were suited to KDD involves the expert's belief about two dimensions: the **open textured - well defined axis** and the **boundedness** dimension.

From our research we have observed that knowledge discovery techniques in law require some manual analysis of the data and the KDD process can only provide support for legal practitioners if commonplace cases are abundant. We also noted that techniques for dealing with contradictions or outliers must be developed for application in the pre-processing phase. In most legal domains, examples are contradictory if they have different outputs given the same input. Contradictions are to be expected in Family Law, since the weighting of factors can vary between judges and even within the same judge over a period of time. For the Split Up project, we considered outputs that differed by a small margin to be contradictions that are allowable. Outputs that differed by a large margin despite identical inputs were considered outliers and labelled extreme.

We dealt with contradictions by first isolating cases that contradicted others to an extreme extent, from those that did not contradict others, or did so only to a minor

extent. Extreme contradictions are interpreted by us as judicial errors, or cases in which factors were taken into account that affected a judgement but were not reported in the judgement.

An approach we did not follow in the Split Up project was to ignore contradictions. If sufficient data is collected, then the majority of typical outputs will outweigh the effects of a handful of extreme cases. Unfortunately the Split Up training set was not large enough to allow us to ignore contradictions.

Our approach was to ignore extreme cases. If two judges, in complete agreement on all relevant facts and legal principles arrive at vastly different outcomes in a discretionary domain then we take the view that at least one outcome is in error and must be removed. This opinion is subjective, since being a discretionary domain, a decision maker is not bound to arrive at the same outcome as expected by others.

We have implemented a degree of consistency in our method for removing extreme cases, by designing a metric that quantifies the extent to which two outcomes are contradictory. This metric relies on the representation of all inputs and outputs as binary digits. Two binary outcomes can be compared by noting the position of the set bit in each outcome. Thus, a binary outcome [1 0 0 0 0] differs from [0 0 0 0 1] by four place units. The set bit in the second number is four places away from the set bit in the former outcome — known as a four place contradiction. Any examples that have identical inputs and also have outputs that differ by three or more bit positions were considered to be extreme and were removed from the training set. Table 1 illustrates the number of contradictions detected and removed for three (of the 20) networks used in the Split Up system. Other networks had levels of contradictions that fell between 0% and 14.56%.

| Network name                     | Description                                                                                       | No. examples | No. of contradictions removed | Training set size |
|----------------------------------|---------------------------------------------------------------------------------------------------|--------------|-------------------------------|-------------------|
| Relative indirect contributions  | Contribution made in an indirect way to the marriage of the husband relative to those of the wife | 103          | 15<br>(14.56%)                | 88                |
| Relative homemaker contributions | Contribution made as a homemaker by the husband relative to those of the wife                     | 103          | 0                             | 103               |
| Individual personal prospects    | Future prospects based on personal skills, abilities age and health                               | 206          | 29<br>(14.07%)                | 177               |

Table 1 — Number of contradictions in the training data for three Split Up neural networks

### 3.3 Data Mining in the Split Up System

The data mining phase in Split Up was performed with neural networks. All networks were trained using 5 fold cross validation. Cross validation is a resampling technique described by [Weiss and Kulikowski 1992] that provides a mechanism for estimating the error rate of a classifier on the true population. The simplest way to define a classifier error is to count the number of examples correctly classified by a neural network.

Counting the number of correctly classified examples leads to a measure of network performance which may be too fine-grained for legal applications. A better measure of a network's performance includes an indication of the magnitude of the error. A variation of 5% either way from a judge's decision of the percentage of assets awarded to the wife, is, in our view, a minor error. A network which outputs a percentage split which deviates from that obtained by a judge by 20% is assumed to have erred. Although the cut off point for declaring that an error has occurred is necessarily subjective, we implement a degree of consistency by comparing the position of the set bit output by the network with that expected from the training set.

Training was halted once the proportion of errors of magnitude 3 (the set bit is 3 or more bit positions away from the expected) was 3% or less. An error of magnitude 3 represents a significant error for most networks. However, the cost of eliminating these errors totally is high in that the additional training required increased the risk of overtraining. An extreme error on only 3% of cases in the true population represents a margin that we considered tolerable in practice. Table 2 illustrates the topology and performance of a sample of networks in Split Up.

| Network name                    | Topology: | No of epochs | Average proportion of errors of magnitude |      |      |      |
|---------------------------------|-----------|--------------|-------------------------------------------|------|------|------|
|                                 |           |              | >3                                        | >2   | >1   | >0.5 |
| Percentage Split                | 15-12-13  | 900          | 0.03                                      | 0.12 | 0.16 | 0.31 |
| Relative contributions          | 20-8-5    | 1130         | 0.01                                      | 0.02 | 0.06 | 0.27 |
| Relative needs                  | 8-3-5     | 230          | 0.00                                      | 0.01 | 0.02 | 0.07 |
| Individual needs                | 14-3-4    | 830          | 0.02                                      | 0.07 | 0.11 | 0.30 |
| Individual personal prospects   | 17-9-5    | 480          | 0.02                                      | 0.01 | 0.05 | 0.26 |
| Individual employment prospects | 14-8-5    | 1170         | 0.02                                      | 0.02 | 0.08 | 0.10 |
| Individual capacity to work     | 12-5-3    | 180          | 0.01                                      | 0.00 | 0.06 | 0.22 |

Table 2 — Performance of Split Up neural networks

## 4 Conclusion

Feature selection algorithms attempt to find an optimal set of features from a larger set. [Vafie and De Jong 1993] stress that it is important to improve methods for automating the process of feature selection to eliminate irrelevant attributes. In some domains, increasing the number of attributes available leads to a degradation in performance. This is particularly the case when there are irrelevant features present in the attribute set; since any noise that is already present in the training examples is magnified by adding more noisy attributes. In this way the presence of additional attributes can interfere with other more useful attributes.

We have applied feature selection methods using genetic algorithms to the data used to determine percentage split in the Split Up system [Skabar *et al* 1997]. An initial set of features is provided to the system as input, in addition to a training set representing examples of the various cases for which classification is to be performed. A genetic algorithm search procedure is then used to explore the space of subsets of the initial feature set. The performance of each feature subset is evaluated by invoking an evaluation function on the classifier induced using the feature subset. Measures of performance include the proportion of correctly classified examples (this is an average over a number of cross validation trials), the complexity of the decision trees induced using ID3 and the size of the subset of features (smaller subsets are preferred). We have used this method successfully to generate a number of feature subsets which lead to improved classification performance as compared with the same induction algorithm trained using all available attributes.

As [Fayyad *et al* 1996c] note the overall KDD process includes the evaluation and possible interpretation of the mined patterns to determine which patterns can be considered new knowledge. In [Zeleznikow and Stranieri 1997a] we discussed our initial evaluation of the knowledge discovery aspects of the Split Up system. Ten domain experts were given three hypothetical cases, and their answers and explanations were compared to the outcomes generated by the Split Up system. Currently Split Up is being beta-tested at other a dozen Melbourne legal firms as well as registries of the Family Court of Australia, offices of the Legal Aid Commission of Victoria and Relationships Australia. Users are asked to complete forms assessing the advice of the Split Up system.

We have discussed knowledge discovery in the legal domain and indicated how the Split Up system uses neural networks to predict judicial outcomes. We noted that even when using data mining, domain expertise is vital. We illustrated that to perform data mining in a legal domain requires an abundance of commonplace cases. Knowledge discovery methods in law must deal with contradictory cases. In Split Up extreme contradictions were removed from the data set to train networks.

Knowledge discovery techniques will not always produce classifiers that are perfect generalisations of all cases. The way we define error and the extent to which error is permitted are issues that impact on KDD in legal domains. In the Split Up system,

we defined a metric for determining the magnitude of error and a subjective heuristic for deciding when the error level is acceptable.

Ninety-four attributes are currently used in the Split Up template. We are currently using genetic algorithms as a feature selection tool to learn how to reduce the number of variables used in the Split Up system.

## 5. References

- Black, H. C. 1990. *BLACK'S LAW DICTIONARY*, West Publishing Company, St. Paul, Minnesota.
- Decker, K. M. and Focardi, S. 1995. *Technology Overview: A Report on Data Mining*. Technical Report 95-02. Swiss Scientific Computing Centre, CSCS-ETH, CH-6928, Manno, Switzerland.
- Edwards, L. and Huntley, A. J. K. 1992. Creating a Civil Jurisdiction Adviser. *Law, Computers and Artificial Intelligence*: 1(1), 5—40.
- Fayyad, U., Piatetsky-Shapiro, G. and Smyth, P. 1996a. From data mining to knowledge discovery: An overview. In *Advances in Knowledge Discovery and Data Mining* AAAI/MIT Press: Cambridge, Massachusetts.
- Fayyad, U., Piatetsky-Shapiro, G. and Smyth, P. 1996b. The KDD Process for Extracting Useful Knowledge from Volumes of Data. *Communications of the ACM*, 39(11): 27-34.
- Fayyad, U., Piatetsky-Shapiro, G. and Smyth, P. 1996c. From Data Mining to Knowledge Discovery in Databases. *AI Magazine*, 17(3) : 37-54.
- Gardner, A. v. d. L. 1987. *An Artificial Intelligence Approach to Legal Reasoning*, Bradford/MIT Press.
- Kolodner, J. 1993. *Case based reasoning*. Los Altos: Morgan Kaufmann.
- Lyons, D. 1985. Formal Justice and Judicial Precedent. *Vanderbilt Law Review*. 38: 495.
- McCarty, L. T. 1977. Reflections on TAXMAN: An Experiment in Artificial Intelligence and Legal Reasoning. *Harvard Law Review* 90: 837.
- Merkl, D. and Schweighofer, D. 1997. The Exploration of Legal Text Corpora with Hierarchical Neural Networks: A Guided Tour in Public International Law. *Proceedings of Sixth International Conference on Artificial Intelligence and Law*, ACM: Melbourne, Australia, 98-105.
- Pannu, A. S. 1995. Using Genetic Algorithms to Inductively Reason with Cases in the Legal Domain. *Proceedings of Fifth International Conference on Artificial Intelligence and Law*, New York, ACM Press:175-184.
- Quinlan, J. R. 1986. Induction of decision trees. *Machine Learning* 1:81-106.
- Rissland, E. L. and Friedmann, M. T. 1995. Detecting Change in Legal Concepts. *Proceedings of Fifth International Conference on Artificial Intelligence and Law*, New York, ACM Press:127-136.
- Skabar, A., Stranieri, A. and Zelezniakow, J. 1997. Using argumentation for the decomposition and classification of tasks for hybrid system development. in Kasabov, N., Kozma, R., Ko, K., O'Shea, R., Coghill, G., and Gedeon, T. (eds)

- Progress in Connectionist Based Information Systems. Proceedings of the 1997 International Conference on Neural Information Processing and Intelligent Information Systems.* Springer-Verlag, Singapore. p814-818.
- Stranieri, A. and Zeleznikow, J. 1992. SPLIT-UP—Expert system to determine spousal property distribution on litigation in the Family Court of Australia. *Proceedings of Artificial Intelligence Conference (Australia)-92*, Hobart: World Scientific, 51-56.
- Stranieri, A., Zeleznikow, J., Gawler, M. and Lewis, B. 1998. A hybrid—neural approach to the automation of legal reasoning in the discretionary domain of family law in Australia. To appear in *Artificial Intelligence and Law*.
- Toulmin, S. 1958. *The uses of argument*. Cambridge: Cambridge University Press.
- Vafie, H and De Jong, K. A. 1993. Robust feature selection in Algorithms. *Proceedings of the International Conference on Tools for Artificial Intelligence*, Boston, Ma., IEEE Computer Society Press: 356-364.
- Weiss, S. and Kulikowski, C. 1992. Computer Systems that Learn: classification and Prediction Methods from Statistics, Neural Nets, Machine Learning and Expert Systems. Morgan Kaufman.
- Wilkins, D. and Pillaipakkamatt, K. 1997. The Effectiveness of Machine Learning Techniques for Predicting Time to Case Disposition. *Proceedings of Sixth International Conference on Artificial Intelligence and Law*, ACM: Melbourne, Australia, 39-46.
- Zeleznikow, J. and Hunter, D. 1994. Building Intelligent Legal Information Systems: Knowledge Representation and Reasoning in Law, Kluwer Computer/Law Series, 13.
- Zeleznikow, J. Hunter, D. and Stranieri, A. 1997. Using cases to build intelligent decision support systems. *Database Applications Semantics — Proceedings of the IFIP Working Group 2.6 Conference*. Stone Mountain, Georgia, USA. May 30 - June 2. 1995. Edited by Meersman, R. and Mark, L. Chapman—Hall: 443-460.
- Zeleznikow, J. and Stranieri, A. 1995. The Split-Up system: Integrating neural networks and rule based reasoning in the legal domain. *Proceedings of Fifth International Conference on Artificial Intelligence and Law*, ACM:185-194.
- Zeleznikow, J. and Stranieri, A. 1997a. Modelling discretion in the Split Up system. *PACIS97 The Pacific Asia Conference on Information Systems*, Information Systems Research Management, Queensland University of Technology: 307-320.
- Zeleznikow, J. and Stranieri, A. 1997b. Knowledge discovery in the Split-Up project. *Proceedings of Sixth International Conference on Artificial Intelligence and Law*, ACM: 89-97.

# Scaling Up the Rule Generation of C4.5

Zijian Zheng

School of Computing and Mathematics  
Deakin University, Geelong  
Victoria 3217, Australia  
(zijian@deakin.edu.au)

**Abstract.** C4.5 is the most well-known inductive learning algorithm. It can be used to build decision trees as well as production rules. Production rules are a very common formalism for representing and using knowledge in many real-world domains. C4.5 generates production rules from raw trees. It has been shown that the set of production rules is usually both simpler and more accurate than the decision tree from which the ruleset was formed. This research shows that generating production rules from pruned trees usually results in significantly simpler rulesets than generating rules from raw trees. This reduction in complexity is achieved without reducing prediction accuracies. Furthermore, the new approach uses significantly less induction time than the latter. This paper uses experiments in a wide variety of natural domains to illustrate these points. It also shows that the new method scales up better than the old one in terms of ruleset size, the number of rules, and learning time when the training set size increases. This is an important characteristic for learning algorithms used for data mining.

## 1 Introduction

Data mining usually deals with very large datasets. In addition to being highly accurate, the mined knowledge is expected to be easy to understand and use. Moreover, learning algorithms for data mining should be relatively fast. Many inductive learning algorithms build decision trees. C4.5 (Quinlan, 1993) is one of the most commonly used decision tree learning algorithms. It is very efficient. C4.5 first builds a large decision tree (called a *raw tree*). It, then, applies pessimistic pruning (Quinlan, 1987b) to prune the raw tree back to overcome the overfitting problem of the raw tree (Quinlan, 1993). The *pruned tree* is smaller and often more accurate than the raw tree (Quinlan, 1987b; 1993). As another method for simplifying raw trees, C4.5 can transform a raw tree into a set of production rules.<sup>1</sup> It has been shown that the ruleset is usually simpler and more accurate than the raw tree from which the ruleset was created (Quinlan, 1987a; 1987b; 1993). Both decision trees and production rules are commonly used to represent and process knowledge in data mining. They are understandable if they

---

<sup>1</sup> C4.5 can transform multiple decision trees for the same task into a set of rules (Quinlan, 1993), but we focus on generating rules from a single tree in this paper.

are not too large. Production rules are even more commonly used in real-world domains.

It is known that the rule generation of C4.5 is relatively slower compared with the tree generation. In addition to the training set size, the computational requirement of the rule generation of C4.5 depends on the size of decision tree from which the ruleset is obtained. Since pruning reduces the sizes of decision trees, it is natural to expect that generating rules from pruned trees needs less time than generating rules from raw trees. One possible problem for creating rules from pruned trees is that the possible rules that can be created from a pruned tree is only a subset of all possible rules that can be created from the corresponding raw tree. As a result, some good rules could be eliminated from the search space of the rule generation by pruning the raw tree. However, pruning only deletes tests whose elimination does not significantly reduce the estimated accuracy of the decision tree (Quinlan, 1993). Therefore, we expect that pruned trees should provide a search space not significantly worse than the space provided by the corresponding raw trees for rule generation in terms of prediction accuracy of production rules to be generated. In addition, we expect that the rulesets derived from pruned trees are simpler than rulesets from the corresponding raw trees.

In this paper, we investigate these issues using experiments in a wide variety of natural domains. We assume that readers are familiar with decision tree learning (Quinlan, 1993; Breiman, Friedman, Olshen, and Stone, 1984). The following section briefly describes how C4.5 transforms a decision tree into a set of production rules, and how to modify C4.5 to make it create rules from pruned trees instead of raw trees. For details about how C4.5 builds decision trees, prunes decision trees, and transforms trees into rules, see Quinlan (1993). Section 3 reports experiments to show the advantages of generating rules from pruned trees over generating rules from raw trees. Finally, Section 4 draws conclusions.

## 2 Generating Rules from Decision Trees

C4.5rules (Quinlan, 1987a) in the C4.5 package (Quinlan, 1993) generates production rules from decision trees. Given a decision tree, C4.5rules derives a set of production rules based on the same training set used when building the tree. The process consists of two stages.

First, individual rules are extracted from a decision tree. Each path from the root of the tree to a leaf is transformed into an initial production rule of the form:

**if**  $Cond_1 \wedge Cond_2 \wedge \dots \wedge Cond_n$  **then class**  $C$ ,

where the  $Cond_i$ 's are conditions of the path and  $C$  is the class labeled by the leaf, called the *predicted class* of the rule. Each such rule is then generalized by deleting conditions that do not seem helpful for discriminating the predicted class from other classes. When generalizing individual rules, some initially distinct rules are simplified into identical rules and other rules are simplified into vacuous rules since all their conditions are eliminated. The number of rules generated in the first stage is generally smaller than the number of leaves in the tree.

In the second stage, all rules are grouped according to their predicted classes. For each class in turn, the set of rules for that class is simplified by deleting rules whose removal does not diminish the accuracy of the set of rules as a whole. After sets of rules for all the classes have been selected, they are ordered to minimize false positive errors (cases that satisfy a rule but really do not belong to its predicted class). Finally, each rule is checked to see whether its removal can actually reduce the errors of the whole ruleset on the training set. If so, the first such rule is deleted, and the ruleset is checked again.

It has been shown that the set of rules is often both less complex and more accurate on unseen cases than the raw tree from which the ruleset is generated (Quinlan, 1987a; 1993). The rule generation technique described above can be applied on either raw trees or pruned trees, although C4.5 creates rules from raw trees. To make C4.5 generate rules from pruned trees, you just need to replace the string “unpruned” with “tree” in the file “genrules.c” of the C4.5 source code.

### 3 Experiments

This section explores the performance of generating rules from pruned trees by comparing with the performance of generating rules from raw trees using a large number of natural domains. The performance measures used here are the error rate on test sets (unseen cases), ruleset (or tree) size, the number of rules in a ruleset, and computational requirement. The size of a tree is the number of decision nodes and leaves in it. The size of a ruleset is the number of rules plus the number of conditions in all the rules of a ruleset. This measure is comparable to the tree size. In the experiments, both raw trees and pruned trees are built using C4.5. They are indicated by C4.5(r) and C4.5(p) respectively. Generating rules from raw trees is performed by using C4.5rules. This is labeled by C4.5rules(r). Generating rules from pruned trees is performed by using a modified C4.5rules. It is exactly the same as C4.5rules except that the modified one uses pruned trees instead of raw trees when creating rules. This is indicated by C4.5rules(p). The computational requirement of building a decision tree, generating rules from a raw tree, and generating rules from a pruned tree is measured using CPU seconds by running C4.5, C4.5rules, and modified C4.5rules respectively on a SUN SPARCstation 5. Note that the release 8 (Quinlan, 1996) of the C4.5 package with its default option settings is used.

#### 3.1 Experimental Domains and Methods

Forty-two natural domains from the UCI machine learning repository (Merz and Murphy, 1997) are used. This test suite covers a wide variety of different domains with respect to dataset size, the number of classes, the number of attributes, and types of attributes.

In every domain, two stratified 10-fold cross-validations (Breiman *et al.*, 1984; Kohavi, 1995) are carried out for each algorithm. The result of each algorithm in each domain reported is an average value over 20 trials. All the algorithms

are run on the same training and test set partitions. To compare two algorithms with respect to a performance measure in a domain, a two-tailed pairwise t-test on the results of the 20 trials is carried out. The difference is considered as significant if the significance level of the t-test is better than 0.05. In tables in the next subsection, ***boldface*** (*italic*) font indicates that C4.5rules(p) is significantly better (worse) than C4.5rules(r).

### 3.2 Experimental Results

Table 1 presents the error rates and sizes of C4.5(r), C4.5(p), C4.5rules(r), and C4.5rules(p), as well as the number of rules of C4.5rules(r) and C4.5rules(p). In terms of error rate, C4.5(p), C4.5rules(r), and C4.5rules(p) provide similar improvements over C4.5(r) on average over the 42 domains, although none of these three improvements is significant at the level of 0.05 using a one-tailed pairwise sign-test on the results in these 42 domains. Comparing C4.5rules(p) with C4.5rules(r) directly, the former is significantly more accurate than the latter in two domains and is significantly less accurate in one domain. However, their average error rates over all these domains are very close. These results indicate that generating rules from decision trees performs similarly to pruning the trees on average for improving prediction accuracy. Furthermore, generating rules from pruned trees has, on average, a similar prediction accuracy to generating rules from raw trees.

From the table, we can see that C4.5(p) reduces tree size of C4.5(r) by half on average over the 42 domains. C4.5rules(r) reduces the size further. C4.5rules(p) achieves the smallest size on average among C4.5(r), C4.5(p), C4.5rules(r), and C4.5rules(p). All these size reductions are significant at the level of 0.05 using the sign-test. Comparing C4.5rules(p) with C4.5rules(r), C4.5rules(p) is significantly smaller than C4.5rules(r) in 19 domains and significantly larger in only one domain. The former is 5% smaller than the latter on average. The one-tailed pairwise sign-test on the ruleset sizes in the 42 domains shows that this difference is significant at a level better than 0.0001. As far as the number of rules is concerned, C4.5rules(p) generates 11% fewer rules than C4.5rules(r) on average. This difference is also significant at a level better than 0.0001 using the sign-test. These results illustrate that transforming trees into rules can reduce theory size more than pruning trees. Creating rules from pruned trees results in significantly smaller rulesets than creating rules from raw trees in terms of both ruleset size and the number of rules.

Table 2 shows the execution time of C4.5 (including both growing a tree and pruning it), C4.5rules(r), and C4.5rules(p). Note that the execution time of C4.5rules(p) given here and in Subsection 3.3 does not include the time for pruning trees, although including it is more appropriate. It is not included for two reasons. First, compared to the execution time for generating rules either from raw trees or from pruned trees, the execution time for pruning trees is very short (see the last line in Table 2) on average. The total time for generating and pruning trees as well as for generating rules from pruned trees (the second column plus the last column in Table 2) is still much shorter, on average, than the

**Table 1.** Error rates (%) and theory complexities of raw trees, pruned trees, rules from raw trees, and rules from pruned trees generated by C4.5

| Domain      | Error rate (%) |      |           |             | Size   |        |           |               | No. of rules |              |
|-------------|----------------|------|-----------|-------------|--------|--------|-----------|---------------|--------------|--------------|
|             | C4.5           |      | C4.5rules |             | C4.5   |        | C4.5rules |               | C4.5rules    |              |
|             | (r)            | (p)  | (r)       | (p)         | (r)    | (p)    | (r)       | (p)           | (r)          | (p)          |
| Annealing   | 5.3            | 7.4  | 6.2       | 5.4         | 144.1  | 76.9   | 49.6      | 51.5          | 20.8         | 21.0         |
| Audiology   | 22.9           | 21.4 | 22.7      | 21.9        | 63.0   | 49.0   | 47.3      | 46.7          | 20.6         | <b>20.0</b>  |
| Automobile  | 17.0           | 16.3 | 20.7      | 21.0        | 78.0   | 60.6   | 42.8      | 42.4          | 24.9         | 24.6         |
| Breast(W)   | 5.4            | 5.1  | 4.6       | 4.3         | 45.0   | 25.7   | 22.8      | <b>20.1</b>   | 8.9          | <b>8.2</b>   |
| KR-KP       | 0.7            | 0.7  | 0.7       | 0.7         | 72.6   | 54.3   | 96.8      | <b>84.0</b>   | 25.8         | <b>21.9</b>  |
| KR-KN       | 7.5            | 8.9  | 6.8       | 7.3         | 81.5   | 44.6   | 63.0      | <b>48.7</b>   | 21.9         | <b>17.1</b>  |
| Credit(A)   | 16.8           | 14.5 | 14.5      | 14.4        | 159.8  | 30.4   | 34.6      | <b>27.0</b>   | 12.6         | <b>8.8</b>   |
| Credit(G)   | 32.5           | 29.4 | 29.3      | 29.6        | 401.1  | 123.9  | 68.8      | <b>50.8</b>   | 23.9         | <b>17.5</b>  |
| Echo        | 39.4           | 37.8 | 37.1      | 37.0        | 17.1   | 9.6    | 11.1      | 10.2          | 4.2          | 4.1          |
| Glass       | 34.1           | 33.6 | 33.6      | 33.1        | 51.2   | 46.0   | 38.2      | 39.2          | 13.4         | 13.6         |
| Heart(C)    | 22.6           | 22.1 | 19.8      | 21.8        | 65.1   | 39.0   | 33.3      | <b>30.1</b>   | 9.9          | 9.3          |
| Heart(H)    | 22.1           | 21.1 | 20.7      | 20.9        | 51.4   | 19.7   | 18.1      | <b>15.3</b>   | 6.8          | <b>5.5</b>   |
| Hepatitis   | 20.6           | 20.6 | 20.5      | 20.2        | 29.4   | 15.9   | 16.6      | <b>12.4</b>   | 8.0          | <b>5.6</b>   |
| Horse colic | 18.1           | 15.7 | 17.1      | 16.6        | 126.0  | 10.4   | 9.5       | 10.8          | 4.8          | 5.0          |
| House votes | 5.6            | 5.6  | 4.8       | 4.8         | 30.1   | 12.8   | 13.1      | <b>9.2</b>    | 6.3          | <b>5.0</b>   |
| Hypo        | 0.5            | 0.5  | 0.5       | 0.5         | 33.1   | 27.6   | 26.1      | <b>24.9</b>   | 10.0         | <b>9.2</b>   |
| Hypothyroid | 0.7            | 0.7  | 0.7       | 0.7         | 30.6   | 12.4   | 15.2      | 13.2          | 6.3          | <b>5.2</b>   |
| Image       | 3.0            | 3.0  | 3.7       | 3.6         | 88.9   | 82.9   | 104.8     | 104.0         | 28.6         | <b>28.0</b>  |
| Iris        | 4.7            | 4.3  | 4.0       | 4.0         | 8.5    | 8.4    | 8.8       | 8.8           | 4.0          | 4.0          |
| Labor       | 22.3           | 23.7 | 18.7      | 23.7        | 13.8   | 6.9    | 6.7       | 5.7           | 3.9          | <b>3.1</b>   |
| LED-24      | 37.8           | 36.5 | 40.2      | <b>38.2</b> | 71.0   | 58.9   | 51.5      | 50.1          | 17.2         | <b>16.1</b>  |
| Letter      | 12.2           | 12.2 | 13.2      | 13.2        | 2563.6 | 2338.2 | 3622.6    | <b>3576.8</b> | 618.4        | <b>605.5</b> |
| Liver       | 35.8           | 35.4 | 32.6      | 33.1        | 58.0   | 49.6   | 37.7      | 37.0          | 12.9         | 12.8         |
| Lung cancer | 52.5           | 57.5 | 58.8      | 60.4        | 25.0   | 19.4   | 5.2       | 5.0           | 4.5          | <b>4.0</b>   |
| Lympho      | 23.2           | 21.9 | 19.2      | 20.9        | 45.5   | 27.2   | 17.6      | 17.4          | 10.3         | <b>9.3</b>   |
| Nettalk(L)  | 24.6           | 25.9 | 30.4      | 30.6        | 6513.4 | 3393.6 | 635.7     | <b>606.2</b>  | 430.6        | <b>388.9</b> |
| Nettalk(P)  | 17.5           | 19.0 | 23.0      | 22.8        | 5260.6 | 2347.3 | 488.9     | <b>433.5</b>  | 329.1        | <b>266.2</b> |
| Nettalk(S)  | 16.6           | 17.3 | 16.8      | 17.1        | 5129.6 | 2053.0 | 381.3     | <b>306.6</b>  | 230.9        | <b>175.7</b> |
| Pima        | 24.4           | 24.0 | 25.0      | 25.0        | 51.8   | 44.9   | 36.1      | <b>34.8</b>   | 10.2         | 9.8          |
| Post        | 41.6           | 29.4 | 32.8      | 28.9        | 33.5   | 1.3    | 8.0       | <b>0.0</b>    | 3.8          | <b>0.0</b>   |
| Tumor       | 59.1           | 59.6 | 59.3      | 59.6        | 118.7  | 78.8   | 54.8      | <b>49.8</b>   | 16.3         | <b>13.4</b>  |
| Promoters   | 15.1           | 17.5 | 12.1      | 15.9        | 33.0   | 22.8   | 11.6      | 10.9          | 7.8          | <b>7.3</b>   |
| Sick        | 1.3            | 1.3  | 1.6       | 1.6         | 69.4   | 49.6   | 36.6      | 35.0          | 12.3         | 11.6         |
| Solar flare | 17.2           | 15.6 | 17.0      | <b>15.6</b> | 170.4  | 5.4    | 36.6      | <b>4.8</b>    | 14.1         | <b>1.8</b>   |
| Sonar       | 26.4           | 26.4 | 26.7      | 26.7        | 26.9   | 26.9   | 25.4      | 25.4          | 8.3          | 8.3          |
| Soybean     | 9.7            | 8.5  | 8.3       | 8.9         | 168.5  | 93.9   | 108.8     | 106.8         | 35.4         | 34.8         |
| Splice      | 8.2            | 5.8  | 6.4       | 6.1         | 415.0  | 170.4  | 215.7     | <b>190.5</b>  | 71.8         | <b>60.4</b>  |
| Tic-Tac-Toe | 12.5           | 13.7 | 1.4       | <b>3.6</b>  | 186.2  | 128.8  | 74.8      | <b>81.0</b>   | 22.6         | <b>25.6</b>  |
| Vehicle     | 28.6           | 28.5 | 27.0      | 26.6        | 154.4  | 139.2  | 98.6      | 98.8          | 27.1         | 27.6         |
| Waveform    | 23.8           | 23.8 | 25.3      | 25.3        | 48.3   | 47.5   | 42.8      | 42.0          | 13.9         | 13.9         |
| Wine        | 9.0            | 9.0  | 8.7       | 8.7         | 10.4   | 10.2   | 12.2      | 12.1          | 4.5          | 4.5          |
| Zoology     | 6.4            | 7.4  | 7.4       | 7.4         | 18.5   | 18.3   | 15.8      | 15.8          | 8.2          | 8.2          |
| average     | 19.2           | 18.8 | 18.6      | 18.8        | 542.0  | 282.9  | 160.6     | 152.3         | 51.8         | 46.2         |

**Table 2.** Computational requirement (CPU seconds) of building trees, generating rules from raw trees, and generating rules from pruned trees

| Domain                    | C4.5 | C4.5rules |               |
|---------------------------|------|-----------|---------------|
|                           |      | (r)       | (p)           |
| Annealing                 | 1.3  | 2.3       | <b>1.2</b>    |
| Audiology                 | 0.4  | 0.5       | <b>0.4</b>    |
| Automobile                | 0.4  | 0.6       | 0.5           |
| Breast cancer (Wisconsin) | 0.3  | 1.2       | <b>0.7</b>    |
| KR-KP (Chess)             | 1.1  | 7.1       | <b>4.8</b>    |
| KR-KN (Chess)             | 0.3  | 1.6       | <b>1.2</b>    |
| Credit (Australia)        | 0.6  | 1.5       | <b>0.6</b>    |
| Credit (Germany)          | 1.0  | 9.1       | <b>2.5</b>    |
| Echocardiogram            | 0.1  | 0.0       | 0.0           |
| Glass                     | 0.4  | 0.4       | 0.3           |
| Heart disease (Cleveland) | 0.4  | 0.4       | 0.8           |
| Heart disease (Hungary)   | 0.3  | 0.4       | <b>0.2</b>    |
| Hepatitis                 | 0.1  | 0.3       | <b>0.1</b>    |
| Horse colic               | 0.4  | 0.6       | <b>0.1</b>    |
| House votes 84            | 0.1  | 0.3       | <b>0.1</b>    |
| Hypo                      | 2.3  | 3.4       | <b>2.8</b>    |
| Hypothyroid               | 2.3  | 3.6       | <b>1.2</b>    |
| Image                     | 8.2  | 7.1       | 7.3           |
| Iris                      | 0.0  | 0.0       | 0.0           |
| Labor                     | 0.0  | 0.0       | 0.0           |
| LED-24                    | 0.1  | 0.2       | 0.2           |
| Letter                    | 76.7 | 3130.9    | <b>2792.6</b> |
| Liver disorders           | 0.2  | 0.5       | 0.6           |
| Lung cancer               | 0.0  | 0.0       | 0.0           |
| Lymphography              | 0.0  | 0.2       | 0.0           |
| Nettalk (Letter)          | 15.9 | 9664.3    | <b>6832.9</b> |
| Nettalk (Phoneme)         | 5.7  | 773.0     | <b>547.3</b>  |
| Nettalk (Stress)          | 2.3  | 222.6     | <b>116.7</b>  |
| Pima Indians diabetes     | 0.7  | 1.0       | 1.3           |
| Postoperative             | 0.0  | 0.0       | 0.0           |
| Primary tumor             | 0.3  | 1.5       | <b>0.9</b>    |
| Promoters                 | 0.1  | 0.1       | 0.1           |
| Sick                      | 4.6  | 5.2       | <b>7.6</b>    |
| Solar flare               | 0.2  | 2.5       | <b>0.1</b>    |
| Sonar                     | 2.2  | 0.7       | 0.7           |
| Soybean                   | 0.6  | 3.4       | <b>1.7</b>    |
| Splice junction           | 2.2  | 29.2      | <b>14.2</b>   |
| Tic-Tac-Toe               | 0.2  | 3.9       | <b>2.5</b>    |
| Vehicle                   | 2.2  | 4.0       | <b>3.6</b>    |
| Waveform 21               | 1.2  | 1.0       | 1.0           |
| Wine                      | 0.2  | 0.1       | 0.1           |
| Zoology                   | 0.0  | 0.0       | 0.0           |
| average                   | 3.2  | 330.6     | 246.4         |

time for generating rules from raw trees (the third column in Table 2). Second, measuring the pruning time along is not so easy, since C4.5 always grows and prunes trees together.

It is very clear that generating rules takes longer than learning decision trees. However, generating rules from pruned trees can reduce the computational requirement of creating rules from raw trees by 25% on average. This reduction is significant at a level of 0.0001 using the sign-test.

### 3.3 Learning Curves

Having studied the error rate, ruleset size, the number of rules, and computational requirement of C4.5rules(p) and C4.5rules(r) using fixed-sized training sets, we move to investigate, using learning curves, the scaling up characteristic of C4.5rules(p) and C4.5rules(r). Here, only six domains with the largest datasets in the test suite are used due to the space limit. They are the Letter, Nettalk(Letter), Nettalk(Phoneme), Nettalk(Stress), Hypo, and Sick domains. Figures 1, 2, 3, and 4 show the error rate, ruleset size, the number of rules, and execution time learning curves of the two algorithms. Each point of a learning curve is an average value over 20 trials. A bar in the figures indicates one standard error on each side of a curve. For each trial, the training set used at every point is a randomly selected subset of the training set used in the corresponding trial of the two 10-fold cross-validations on the entire dataset of the domain. In each trial, the training set at a point is a proper subset of the training set at the next adjacent point. The test set at every point of a trial is the same as the test set used in the corresponding trial of the two 10-fold cross-validations.

From Figure 1, we can see that C4.5rules(p) and C4.5rules(r) perform very similarly in terms of error rate when the training set size increases in all the six domains. In the Nettalk(Phoneme) domain, C4.5rules(p) is less accurate than C4.5rules(r) when training sets are small, but the former catches up with the latter after the training set size is over 3000.

As far as ruleset size is concerned, Figure 2 shows that C4.5rules(p) always has smaller rulesets and its ruleset sizes increase less rapidly than C4.5rules(r) in all the six domains. The same trend is illustrated for the number of rules in Figure 3.

Figure 4 clearly shows in five out of the six domains that the computational requirement of C4.5rules(p) is less than that of C4.5rules(r) at nearly all training set sizes. The rate of increase in execution time as training set size increases is lower for C4.5rules(p) than for C4.5rules(r). The Sick domain is an exception. In this domain, C4.5rules(p) takes more time than C4.5rules(r) after the training set size exceeds 2400. The reason is as follows. We have mentioned before that the rule generation of C4.5 consists of two steps: creating individual rules from paths of a tree, and finding a subset of rules from rules derived in the first step based on the MDL principle (Quinlan, 1993). The second step is usually more time-consuming than the first step. In the second step, for each class, C4.5 searches for the best subset of rules from those rules that predict this class. If the number of initial rules for the class is less than or equal to 10, C4.5 uses

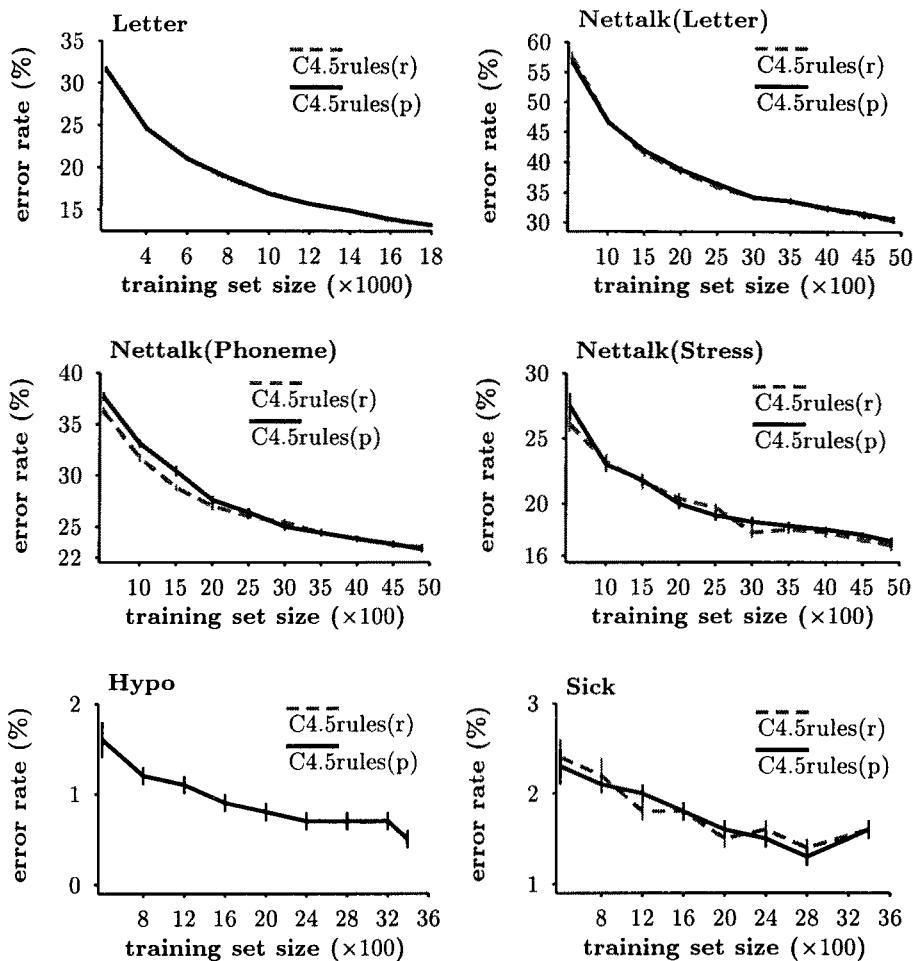


Fig. 1. Error rate curves

exhaustive search. Otherwise, it uses a series of greedy searches (Quinlan, 1995). In the Sick domain, when the training set size is larger than 2400, the number of rules for a single class generated by C4.5rules(p) in the first step is often just less than 10, while it is often just more than 10 for C4.5rules(r). Therefore, in the second step, C4.5rules(p) often uses the exhaustive search, and C4.5rules(r) often uses the greedy search. In consequence, C4.5rules(p) takes longer than C4.5rules(r), as illustrated in Figure 4 (the bottom-right graph).

To verify this explanation, we modify C4.5rules(r) and C4.5rules(p) by forcing them to use the greedy search in the second step regardless of the number of initial rules. Figure 5 (a) shows the execution time of C4.5rules(r) and C4.5rules(p) after this modification in the Sick domain. It is very clear that

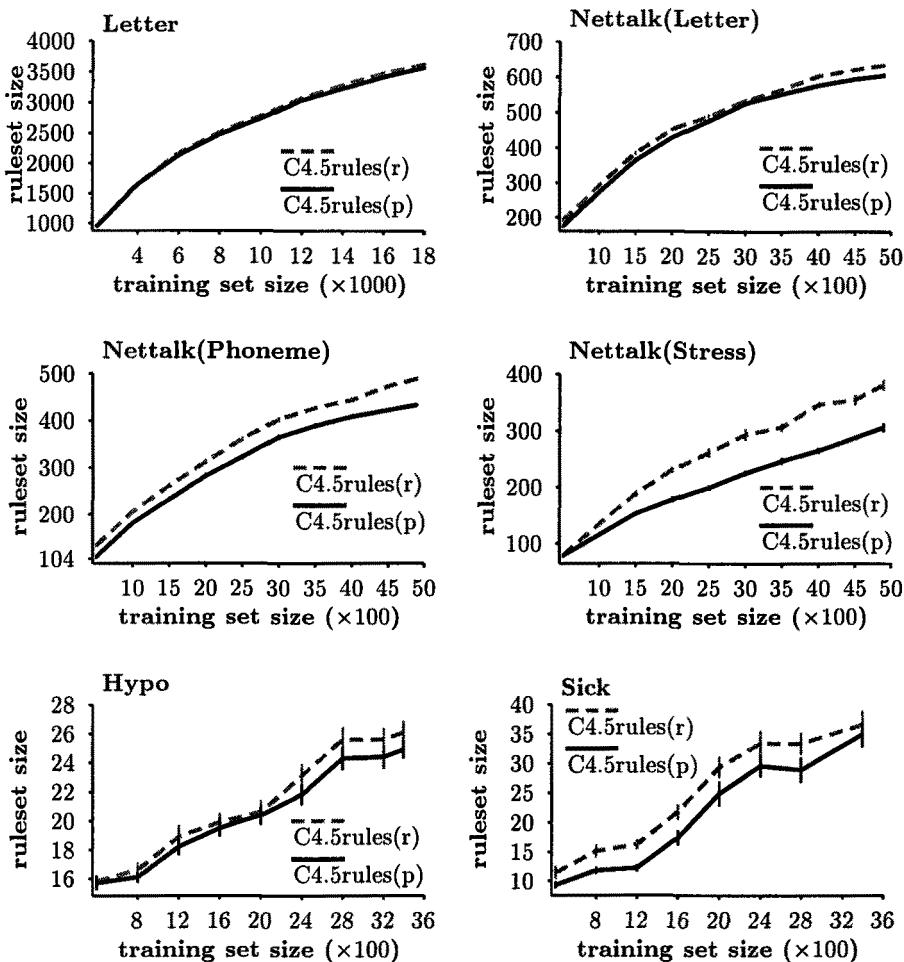


Fig. 2. Ruleset size curves

C4.5rules(r) uses more time than C4.5rules(p) at all training set sizes after 800. Moreover, their difference becomes larger and larger as the training set size increases. If we force C4.5rules(r) and C4.5rules(p) to use exhaustive search in the second step even if the number of initial rules for a class is more than 10, the execution time of C4.5rules(r) and C4.5rules(p) is as in Figure 5 (b). Again, C4.5rules(r) takes longer than C4.5rules(p) after the training set size 800. With this modification, C4.5rules(r) uses much more time than C4.5rules(p) after the training set size exceeds 2400. Note that both of these modifications only affect the accuracy, ruleset size, and the number of rules of C4.5rules(r) and C4.5rules(p) very slightly in this domain.

In summary, the figures illustrate the clear advantages of generating rules

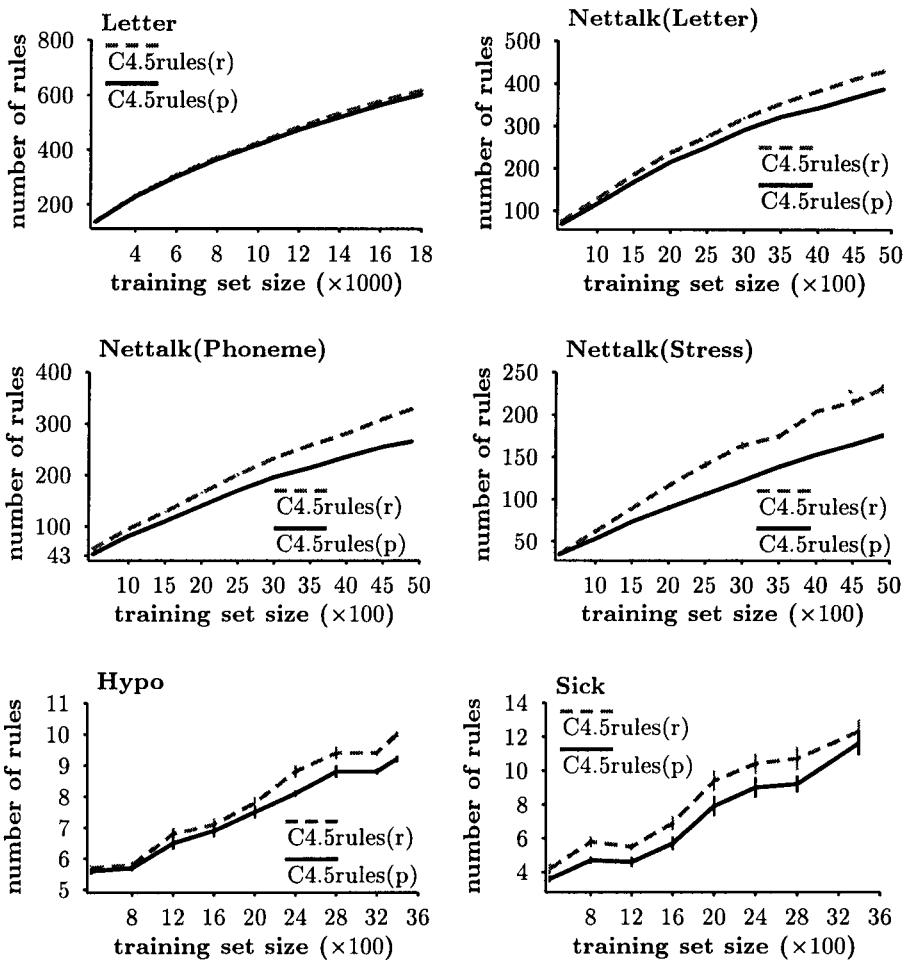


Fig. 3. Rule number curves

from pruned trees over generating rules from raw trees in terms of ruleset size, the number of rules, and computational requirement, while they have similar accuracy performance.

## 4 Conclusions

This paper empirically explored generating production rules from pruned trees by comparing with generating rules from raw trees using a wide variety of natural domains. The latter is adopted by the C4.5 system. The experimental results show that generating rules from pruned trees and generating rules from raw trees result in rulesets with similar overall prediction accuracies. However, the rulesets

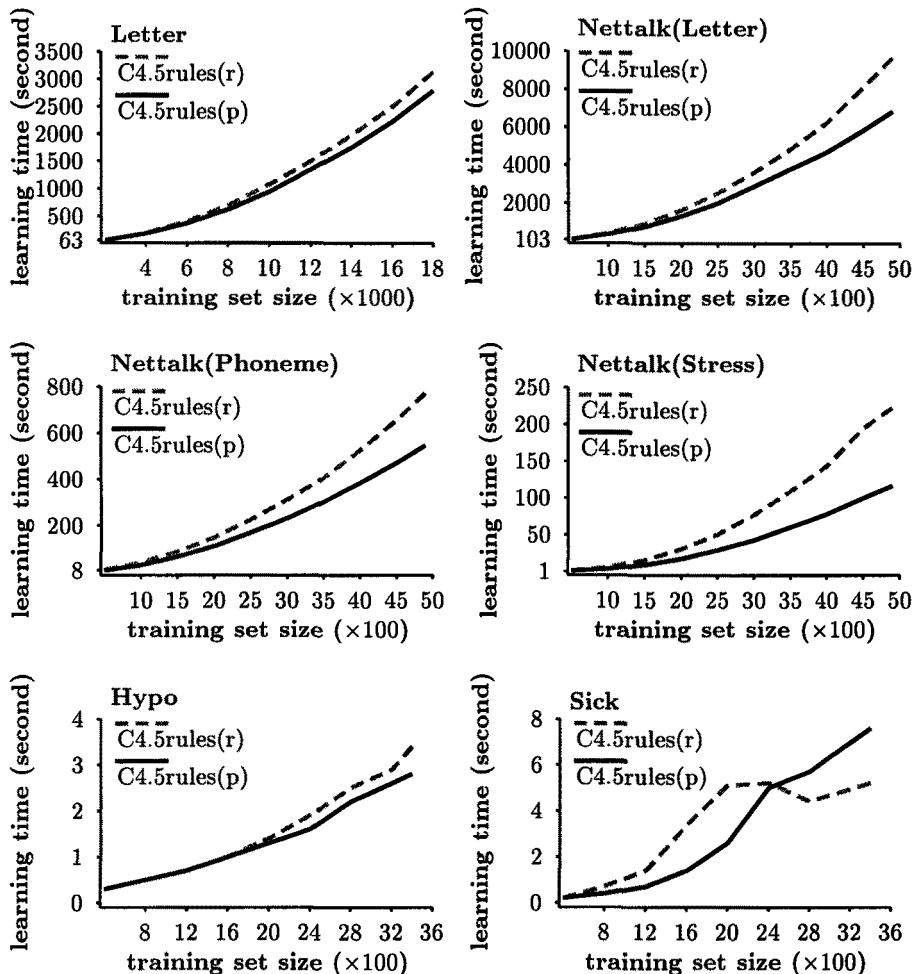
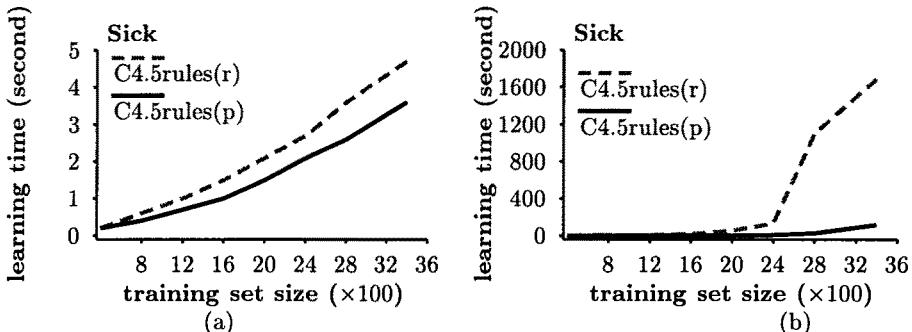


Fig. 4. Computational requirement

created by the former are, on average, significantly smaller than those created by the latter in terms of both ruleset size and the number of rules. Furthermore, the former requires, on average, 25% less execution time than the latter. The experimental study also indicates that while generating rules from pruned trees and generating rules from raw trees scale up similarly well in terms of prediction accuracy, the former scales up often significantly better than the latter in terms of ruleset size, the number of rules, and computational requirement. Therefore, we strongly recommend using pruned trees rather than raw trees when using C4.5 to generate production rules for data mining.



**Fig. 5.** Computational requirement of the rule generation; (a) using a series of greedy searches for rule subset selection, (b) using exhaustive search for rule subset selection.

## Acknowledgements

The author is grateful to Geoffrey Webb for his helpful comments on this research and earlier drafts of the paper. Many thanks to Ross Quinlan for providing C4.5.

## References

- Breiman, L., Friedman, J.H., Olshen, R.A., and Stone, C.J.: *Classification And Regression Trees*. Belmont, CA: Wadsworth (1984).
- Kohavi, R.: A study of cross-validation and bootstrap for accuracy estimation and model selection. *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*. San Mateo, CA: Morgan Kaufmann (1995) 1137-1143.
- Merz, C.J. and Murphy, P.M.: UCI Repository of Machine Learning Databases [<http://www.ics.uci.edu/~mlearn/MLRepository.html>]. Irvine, CA: University of California, Department of Information and Computer Science (1997).
- Quinlan, J.R.: Generating production rules from decision trees. *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*. San Mateo, CA: Morgan Kaufmann (1987a) 304-307.
- Quinlan, J.R.: Simplifying decision trees. *International Journal of Man-Machine Studies* **27** (1987b) 221-234.
- Quinlan, J.R.: *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann (1993).
- Quinlan, J.R.: MDL and categorical theories (continued). *Proceedings of the Twelfth International Conference on Machine Learning*. San Francisco, CA: Morgan Kaufmann (1995) 464-470.
- Quinlan, J.R.: Improved use of continuous attributes in C4.5. *Journal of Artificial Intelligence Research* **4** (1996) 77-90.

# Data Mining Based on the Generalization Distribution Table and Rough Sets

Ning Zhong<sup>1</sup>, Juzhen Dong<sup>1</sup>, and Setsuo Ohsuga<sup>2</sup>

<sup>1</sup> Dept. of Computer Science and Sys. Eng., Yamaguchi University

<sup>2</sup> Dept. of Information and Computer Science, Waseda University

**Abstract.** This paper introduces a new approach for mining *if-then* rules in databases with uncertainty and incompleteness. This approach is based on the combination of *Generalization Distribution Table (GDT)* and the *rough set* methodology. The GDT provides a probabilistic basis for evaluating the strength of a rule. It is used to find the rules with larger strengths from possible rules. Furthermore, the rough set methodology is used to find minimal relative reducts from the set of rules with larger strengths. The strength of a rule represents the uncertainty of the rule, which is influenced by both unseen instances and noises. By using our approach, a minimal set of rules with larger strengths can be acquired from databases with noisy, incomplete data. We have applied this approach to discover rules from some real databases.

## 1 Introduction

Data mining and knowledge discovery is becoming an important topic in AI and is attracting the attention of leading researchers in databases. Many researchers have investigated the inductive methods for knowledge discovery from databases [2]. Among the methods, *version-space* is a typical bottom-up, incremental one, in which learning a concept is possible not only when instances are input simultaneously but also when they are given one by one [5, 6]. However, it is difficult to handle noisy and incomplete data, and it is weak in mining rules from very large, complex databases. On the other hand, the *rough set* methodology for data mining introduced by Pawlak is well known for its ability to acquire decision rules from noisy, incomplete data [12, 11].

This paper introduces a new approach for mining *if-then* rules in databases with uncertainty and incompleteness. The approach combines the Generalization Distribution Table (GDT) with the rough set methodology. The GDT is a table in which the probabilistic relationships between concepts and instances over discrete domains are represented. It is an extension of version-space as a hypothesis search space for generalization. By using a GDT as a hypothesis search space and combining the GDT with the rough set methodology, noisy data and unseen instances can be handled, biases can be flexibly selected, background knowledge can be used to constrain rule generation, and *if-then* rules with strengths can be effectively acquired from large, complex databases. In this paper, we first outline the rough set methodology for rule discovery. Then we define the GDT. Further-

more, we describe on basic concepts and an implementation of our methodology, and an application.

## 2 The Rough Set Methodology

In the rough set methodology to data mining, a database is regarded as a decision table, which is denoted  $T = (U, A, C, D)$ , where  $U$  is universe of discourse,  $A$  is a family of equivalence relations over  $U$ , and  $C, D \subset A$  are two subsets of attributes that are called condition and decision attributes, respectively [12].

The process of data mining is that of simplifying a decision table and generating minimal decision algorithm. In general, an approach for decision table simplification consists of the following steps:

1. Computation of reducts of condition attributes that is equivalent to elimination of some column from the decision table.
2. Elimination of duplicate rows.
3. Elimination of superfluous values of attributes.

A representative approach for the problem of reducts of condition attributes is the one to represent knowledge in the form of a discernibility matrix [7, 12]. The basic idea can be briefly presented as follows:

Let  $T = (U, A, C, D)$  be a decision table, with  $U = \{u_1, u_2, \dots, u_n\}$ . By a *discernibility matrix* of  $T$ , denoted  $M(T)$ , we will mean  $n \times n$  matrix defined thus:

$$m_{ij} = \{a \in A : a(u_i) \neq a(u_j)\} \text{ for } i, j = 1, 2, \dots, n.$$

Thus entry  $m_{ij}$  is the set of all attributes that discern objects  $u_i$  and  $u_j$ .

Generating minimal decision algorithm is to eliminate the superfluous decision rules associated with the same decision class. It is obvious that some decision rules can dropped without disturbing the decision-making process, since some other rules can take over the job of the eliminated rules.

## 3 Generalization Distribution Table

The central idea of our methodology is to use *Generalization Distribution Table (GDT)*, as a hypothesis search space for generalization, in which the probabilistic relationships between concepts and instances over discrete domains are represented [13, 14, 15]. A GDT is a table that consists of three components: the possible instances, the possible generalizations for instances, and the probabilistic relationships between the possible instances and the possible generalizations.

The *possible instances*, which are denoted in columns in a GDT, are all possible combinations of attribute values in a database. The number of the possible instances is

$$\prod_{i=1}^m n_i, \quad (1)$$

where  $m$  is the number of attributes,  $n_i$  is the number of different data values in attribute  $i$ .

The *possible generalizations* for instances, which are denoted in rows in a GDT, are all possible generalizations for all possible instances, and the number of the possible generalizations is

$$\prod_{i=1}^m (n_i + 1) - \left( \prod_{i=1}^m n_i \right) - 1. \quad (2)$$

The *probabilistic relationships* between the possible instances and the possible generalizations, which are denoted as the elements  $t_{ij}$  in a GDT, are the probabilistic distributions for describing the strength of the relationship between every possible instance and every possible generalization.

If we do not use any prior background knowledge, the prior probability distributions are equiprobable and are defined by Eq. (3),

$$p(PI_j | PG_i) = \begin{cases} \frac{1}{N_{PG_i}} & \text{if } PI_j \supset PG_i \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

where  $PI_j$  is the  $j$ th possible instance,  $PG_i$  is the  $i$ th possible generalization, and  $N_{PG_i}$  is the number of the possible instances satisfying the  $i$ th possible generalization, that is,

$$N_{PG_i} = \prod_j^m n_j, \quad (4)$$

where  $j = 1, \dots, m$ , and  $j \neq$  the attribute that is contained by the  $i$ th possible generalization (i.e.,  $j$  just contains the attributes expressed by the wild card as shown in Table 1).

Furthermore, background knowledge can be used as a bias to constrain the possible instances and the prior probabilistic distributions. For example, if we use a background knowledge,

*"when the air temperature is very high, it is not possible there exists some frost at ground level"*,

then we do not consider the possible instances that are contradictory with this background knowledge in all possible combination of different attribute values in a database for creating a GDT. Thus, we can get the more refined result by using background knowledge in the learning process to be stated in Section 5.

Here we define the strength of a generalization as follows:

$$s(PG_i) = \frac{N_{ins-rel,i}}{N_{PG_i}} = \sum_j (p(PI_j | PG_i)) \quad (5)$$

where  $N_{ins-rel,i}$  is the number of the observed instances satisfying the  $i$ th generalization. The initial value of  $s(PG_i)$  is 0. The value will be dynamically updated according to the real data in a database. If all of the instances satisfying  $i$ th generalization appear, the strength will be the maximal value, 1. The larger the value of  $s(PG_i)$ , the more strong the  $i$ th generalization.

**Table 1.** The Generalization Distribution Table for a sample database shown in Table 2  
(Note: in the GDT, the elements that are not displayed are all zero.)

|       | a0b0c0 | a0b0c1 | a0b1c0 | a0b1c1 | a0b2c0 | a0b2c1 | a1b0c0 | a1b0c1 | a1b2c1 | ... |
|-------|--------|--------|--------|--------|--------|--------|--------|--------|--------|-----|
| *b0c0 | 1/2    |        |        |        |        |        | 1/2    |        |        | ... |
| *b0c1 |        | 1/2    |        |        |        |        |        | 1/2    |        | ... |
| *b1c0 |        |        | 1/2    |        |        |        |        |        |        | ... |
| *b1c1 |        |        |        | 1/2    |        |        |        |        |        | ... |
| *b2c0 |        |        |        |        | 1/2    |        |        |        |        | ... |
| *b2c1 |        |        |        |        |        | 1/2    |        |        |        | 1/2 |
| a0*c0 | 1/3    |        | 1/3    |        | 1/3    |        |        |        |        | ... |
| ...   |        |        |        |        |        |        |        |        |        | ... |
| *c0   | 1/6    |        | 1/6    |        | 1/6    |        | 1/6    |        |        | ... |
| *c1   |        | 1/6    |        | 1/6    |        | 1/6    |        | 1/6    |        | 1/6 |
| *b0*  | 1/4    | 1/4    |        |        |        |        | 1/4    | 1/4    |        | ... |
| *b1*  |        |        | 1/4    | 1/4    |        |        |        |        |        | ... |
| *b2*  |        |        |        |        | 1/4    | 1/4    |        |        |        | 1/4 |
| a0**  | 1/6    | 1/6    | 1/6    | 1/6    | 1/6    | 1/6    |        |        |        | ... |
| a1**  |        |        |        |        |        |        | 1/6    | 1/6    |        | 1/6 |

**Table 2.** A sample database

| No | a  | b  | c  | d |
|----|----|----|----|---|
| u1 | a0 | b0 | c1 | y |
| u2 | a0 | b1 | c1 | y |
| u3 | a0 | b0 | c1 | y |
| u4 | a1 | b1 | c0 | n |
| u5 | a0 | b0 | c1 | n |
| u6 | a0 | b2 | c1 | n |
| u7 | a1 | b1 | c1 | y |

Table 1 is an example of the GDT created for a sample database as shown in Table 2, in which three attributes,  $a, b, c$ ,

$$a \in \{a_0, a_1\}, b \in \{b_0, b_1, b_2\}, c \in \{c_0, c_1\},$$

are used. Thus, the number of the possible instances for this example is 12. Furthermore, “\*” in Table 1, which specifies a wild card, denotes the generalization for instances. For example, the generalization {\*b0c1} for the instance {a0b0c1} means the attribute  $a$  is unimportant for describing a concept. And the number of the possible generalizations is 23.

It merits our attention that Eq. (5) is not suitable for duplicate instances. Hence the duplicate instances should be handled before using this equation.

## 4 Data Mining based on the GDT and Rough Sets

Based on the preparation in the above sections, this section describes the basic methodology of mining *if-then* rules, which is based on the combination of the GDT and rough sets.

### 4.1 Rule Representation and Condition/Decision Attributes

Let  $T = (U, A, C, D)$  be a decision table,  $U$  a universe of discourse,  $A$  a family of equivalence relations over  $U$ , and  $C, D \subset A$  two subsets of attributes that are called condition and decision attributes, respectively.

The learned rules are typically expressed in

$$X \rightarrow Y \text{ with } S. \quad (X \in C, Y \in D)$$

That is, “a rule  $X \rightarrow Y$  has a strength  $S$  in a given decision table  $T$ ”. Where  $X$  denotes the conjunction of the conditions that a concept must satisfy,  $Y$  denotes a concept that the rule describes, and  $S$  is a “measure of strength” of which the rule holds.

The example shown in Table 2 is, in fact, a decision table in which  $U = \{u_1, u_2, \dots, u_7\}$ , condition attributes  $C = \{a, b, c\}$ , a decision attribute  $D = \{d\}$ .

Usually, the decision attributes are not used to create the GDT, but are used to distinguish contradictory rules and different concepts (classes).

### 4.2 Rule Strength

We define the strength  $S$  of a rule  $X \rightarrow Y$  in a given decision table  $T$  as follows:

$$S(X \rightarrow Y) = s(X) \times (1 - r(X \rightarrow Y)). \quad (6)$$

From Eq. (6) we can see that the strength  $S$  of a rule is affected by the following two factors:

1. The strength of the generalization  $X$  (i.e., the condition of the rule),  $s$ . It is given by Eq. (5).
2. The rate of noises,  $r$ . It shows the quality of classification, that is, how many instances as the conditions that a rule must satisfy can be classified into some class.

$$r(X \rightarrow Y) = \frac{N_{ins-rel}(X) - N_{ins-class}(X, Y)}{N_{ins-rel}(X)}, \quad (7)$$

where  $N_{ins-rel}(X)$  is the number of the observed instances satisfying the generalization  $X$ ,  $N_{ins-class}(X, Y)$  is the number of the instances belonging to the class  $Y$  within the instances satisfying the generalization  $X$ .

From the GDT, we can see that a generalization is 100% true if and only if all of instances belonging to this generalization appear. Let us again use the example shown in Table 2. Considering the generalization  $\{a_0b_1\}$ , if instances both  $\{a_0b_1c_0\}$  and  $\{a_0b_1c_1\}$  appear, the strength  $s(\{a_0b_1\})$  is 1; if only one of  $\{a_0b_1c_0\}$  and  $\{a_0b_1c_1\}$  appears, the strength  $s(\{a_0b_1\})$  is 0.5, as shown in Figure 1. We can see that both  $\{a_0b_1\}$  and  $\{b_1c_1\}$  are generalizations for the instance  $\{a_0b_1c_1\}$ . But the strengths of them are  $s(\{a_0b_1\}) = 0.5$  and  $s(\{b_1c_1\}) = 1$ , respectively. No matter what value of noise rate  $r$  may be, the strength of the rule  $\{b_1c_1\} \rightarrow y$  is greater than the strength of the rule  $\{a_0b_1\} \rightarrow y$ .

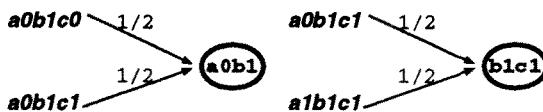


Fig. 1. Probability of a generalization rule

If a generalization contains the instances belonging to different decision classes, the rule acquired from the generalization is noisy. Furthermore, when the value of noise is over the threshold, the rule is contradictory. As the example in Table 2, the generalization  $\{a_1b_1\}$  is such a contradictory generalization. Due to the strength  $s(\{a_1b_1\}) = 1$  and the noise rates for decision classes  $y$  and  $n$  are  $\frac{1}{2}$ ,  $S(a_1 \wedge b_1 \rightarrow y) = S(a_1 \wedge b_1 \rightarrow n) = 0.5$ .

A user can specify an allowed noise rate as the threshold value. Thus, the rules with the larger rates than the threshold value will be deleted.

### 4.3 Simplifying a Decision Table by Using the GDT

By using the GDT, it is obvious that one instance can be expressed by several possible generalizations, and several instances can be also expressed by one possible generalization. Simplifying a decision table is to find such a set of generalizations, which cover all of the instances in a decision table and the number of generalizations is minimal.

The method of computing the reducts of condition attributes in our approach, in principle, is equivalent to the discernibility matrix method [7, 12], but we do not remove dispensable attributes. This is because

- The greater the number of dispensable attributes, the more difficult it is to acquire the best solution;
- Some values of a dispensable attribute may be indispensable for some values of a decision attribute.

Figure 2.(1) gives the relationship among generalizations. We can see that every generalization in upper levels contains all generalizations related to it in lower levels. That is,

$$\{a_0\} \supset \{a_0b_1\}, \{a_0c_1\} \supset \{a_0b_1c_1\}.$$

In other words,  $\{a_0\}$  can be specialized into  $\{a_0b_1\}$  and  $\{a_0c_1\}$  only. In contrast,  $\{a_0b_1\}$  and  $\{a_0c_1\}$  can be generalized into  $\{a_0\}$ . If the rule  $\{a_0\} \rightarrow y$  is true, the rules  $\{a_0b_1\} \rightarrow y$  and  $\{a_0c_1\} \rightarrow y$  are also true.

It is clear that if a generalization for some instances is contradictory, the related generalizations in upper levels than this generalization are also contradictory. As shown in Figure 2.(2),  $\{a_0c_1\}$  is a contradictory generalization for the instance  $\{a_0b_1c_1\}$ , so that the generalizations  $\{a_0\}$  and  $\{c_1\}$  are also contradictory. Hence, for the instance  $\{a_0b_1c_1\}$ , the generalizations  $\{a_0c_1\}$ ,  $\{b_1\}$ ,  $\{a_0\}$ , and  $\{c_1\}$  are contradictory. Thus, only the generalizations  $\{a_0b_1\}$  and  $\{b_1c_1\}$  can be used.

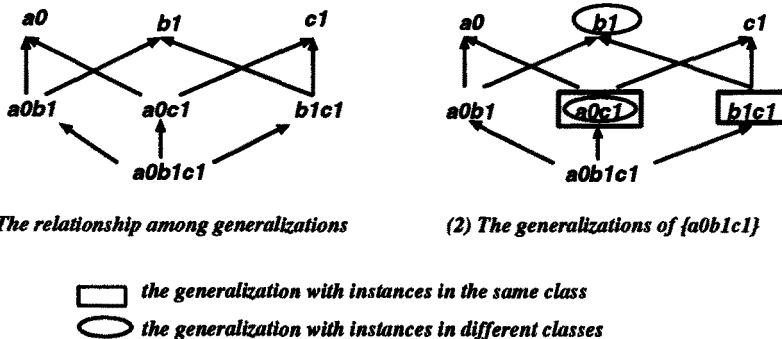


Fig. 2. The generalizations

This result is the same as the one of the discernibility matrix method when no noise exists in the database [7, 12]. Let  $G_-$  be contradictory generalizations,  $G_T$  be all possible consistent generalizations obtained from a discernibility matrix. Clearly,  $G_T = \overline{G_-}$ . That is,

$$G_T = \{b_1\} \cap (\{a_0\} \cup \{c_1\}) = \{b_1a_0\} \cup \{b_1c_1\}$$

$$\begin{aligned} \overline{G_-} &= \overline{\{a_0c_1\} \cup \{b_1\}} = \overline{\{a_0c_1\}} \cap \overline{\{b_1\}} = \{b_1\} \cap (\{a_0\} \cup \{c_1\}) \\ &= \{b_1a_0\} \cup \{b_1c_1\}. \end{aligned}$$

For the database with noises, the generalization that contains instances with different classes should be checked. If a generalization contains more instances

belonging to a class than those belonging to other classes, and the noise rate is smaller than a threshold value, the generalization is regarded as a consistent generalization of that class. Otherwise, the generalization is contradictory. Furthermore, if two generalizations in the same level have different strengths, the one with larger strength will be selected first.

#### 4.4 Rule Selection

There are several possible ways for rule selection. For example,

- Select the rules that contain as many instances as possible;
- Select the rules in the levels as high as possible according to the first type of biases stated above;
- Select the rules with larger strengths.

Here we would like to describe a method of rule selection for our purpose as follows:

- Since our purpose is to simplify the decision table, the rules that contain less instances will be deleted if a rule that contains more instances exists.
- Since we prefer simpler results of generalization (i.e., more general rules), we first consider the rules corresponding to an upper level of generalization.
- The rules with larger strengths are first selected as the real rules.

### 5 The Learning Process

Since mining rules can be viewed as a process of generalization, the learning process is that of choosing the best generalization rules. We first find the contradictory rules and delete them, and then select the rules with larger strengths as the rules mined. The main steps of the learning process are as follows:

*Step 1.* Create one or more GDTs.

In fact, this step can be omitted because the prior distribution of a generalization can be calculated by Eq. (3) and Eq. (4), if we do not use any prior background knowledge for this calculation.

*Step 2.* Handle duplicate instances as shown in the first three columns in Table 3, so that the probability of generalization can be calculated correctly.

*Step 3.* For each instance, find out all generalizations with other instances, and then divide all of the generalizations into two sets  $G_+$  and  $G_-$ . Here  $G_+$  contains all consistent generalizations, and  $G_-$  contains all contradictory generalizations.

Let current instance belong to class  $y$ . The generalizations with class  $y$  are consistent and belong to  $G_+$ . The generalizations with the different class from  $y$  are contradictory and belong to  $G_-$ . If a generalization

contains the instances belonging to different classes, it belongs to  $G_+$  only if the noise rate belonging to the class  $y$  is less than the threshold. Figure 2.(2) shows a result of generalizing for the instance  $\{a_0b_1c_1\}$ . That is,

$$G_+ = \{b_1c_1\}, \quad G_- = \{\{a_0c_1\}, \{b_1\}\}.$$

*Step 4.* Acquire the consistent generalizations by  $\overline{G}_-$ , and the result is denoted in  $G_T$ . That is,  $G_T = \overline{G}_-$ .

*Step 5.* Revise the strength of the generalizations in  $G_T$  that contains one of the generalizations in  $G_+$  in Eq. (5).

*Step 6.* Select the generalizations by using one of the methods stated in Section 4.4 as the mined rules corresponding to some instances.

For example, if we use the method of selecting the rules with larger strengths, the unique candidate is  $\{b_1c_1\}$  for the instance  $\{a_0b_1c_1\}$ . Thus the rule is  $\{b_1c_1\} \rightarrow y$  with  $S = 1$ .

*Step 7.* Go back to *Step 3* until all of instances are handled.

Let the threshold value for the noise rate be 0, the learning result for the sample database shown in Table 2 is shown in Table 3.

**Table 3.** Rules learned from the sample database shown in Table 2

| No       | abc    | d     | $G_T$     | rules                     |
|----------|--------|-------|-----------|---------------------------|
| u1,u3,u5 | a0b0c1 | y,y,n | -         | -                         |
| u2       | a0b1c1 | y     | a0c1,b1c1 | b1c1 $\rightarrow$ y, 1   |
| u7       | a1b1c1 | y     | c1, b1c1  | b1c1 $\rightarrow$ y, 1   |
| u4       | a1b1c0 | n     | c0        | c0 $\rightarrow$ n, 0.167 |
| u6       | a0b2c1 | n     | b2        | b2 $\rightarrow$ n, 0.25  |

## 6 An Application

Some of databases such as postoperative patient, earthquack, weather, mushroom, cancer have been tested for our approach. We would like to use the mushroom database as an example [9].

### 6.1 The Data

The mushroom database includes descriptions of hypothetical samples corresponding to 23 species of gilled mushrooms in the Agaricus and Lepiota Family. Each species is identified as definitely edible, definitely poisonous, or of unknown edibility and not recommended. This latter class was combined with the poisonous one. The guide clearly states that there is no simple rule for determining

the edibility of a mushroom; no rule like “leaflets three, let it be” for Poisonous Oak and Ivy.

1. Number of Instances: 8124

2. Number of Attributes: 22 (all nominally valued)

3. Attribute Information: (classes: edible=e, poisonous=p)

|                               |                                                                                                                    |
|-------------------------------|--------------------------------------------------------------------------------------------------------------------|
| 1. cap-shape:                 | bell=b, conical=c, convex=x, flat=f,<br>knobbed=k, sunken=s                                                        |
| 2. cap-surface:               | fibrous=f, grooves=g, scaly=y, smooth=s                                                                            |
| 3. cap-color:                 | brown=n, buff=b, cinnamon=c, gray=g, green=r,<br>pink=p, purple=u, red=e, white=w, yellow=y                        |
| 4. bruises:                   | bruises=t, no=f                                                                                                    |
| 5. odor:                      | almond=a, anise=l, creosote=c, fishy=y, foul=f,<br>musty=m, none=n, pungent=p, spicy=s                             |
| 6. gill-attachment:           | attached=a, descending=d, free=f, notched=n                                                                        |
| 7. gill-spacing:              | close=c, crowded=w, distant=d                                                                                      |
| 8. gill-size:                 | broad=b, narrow=n                                                                                                  |
| 9. gill-color:                | black=k, brown=n, buff=b, chocolate=h, gray=g,<br>green=r, orange=o, pink=p, purple=u, red=e,<br>white=w, yellow=y |
| 10. stalk-shape:              | enlarging=e, tapering=t                                                                                            |
| 11. stalk-root:               | bulbous=b, club=c, cup=u, equal=e,<br>rhizomorphs=z, rooted=r, missing=?                                           |
| 12. stalk-surface-above-ring: | fibrous=f, scaly=y, silky=k, smooth=s                                                                              |
| 13. stalk-surface-below-ring: | fibrous=f, scaly=y, silky=k, smooth=s                                                                              |
| 14. stalk-color-above-ring:   | brown=n, buff=b, cinnamon=c, gray=g, orange=o,<br>pink=p, red=e, white=w, yellow=y                                 |
| 15. stalk-color-below-ring:   | brown=n, buff=b, cinnamon=c, gray=g, orange=o,<br>pink=p, red=e, white=w, yellow=y                                 |
| 16. veil-type:                | partial=p, universal=u                                                                                             |
| 17. veil-color:               | brown=n, orange=o, white=w, yellow=y                                                                               |
| 18. ring-number:              | none=n, one=o, two=t                                                                                               |
| 19. ring-type:                | cobwebby=c, evanescent=e, flaring=f, large=l,<br>none=n, pendant=p, sheathing=s, zone=z                            |
| 20. spore-print-color:        | black=k, brown=n, buff=b, chocolate=h, green=r,<br>orange=o, purple=u, white=w, yellow=y                           |
| 21. population:               | abundant=a, clustered=c, numerous=n,<br>scattered=s, several=v, solitary=y                                         |
| 22. habitat:                  | grasses=g, leaves=l, meadows=m, paths=p,<br>urban=u, waste=w, woods=d                                              |

4. Missing Attribute Values: 2480 of them (denoted by "?"), all for attribute #11.

5. Class Distribution:

- edible: 4208 (51.8%)

- poisonous: 3916 (48.2%)
- total: 8124 instances

## 6.2 The Result

In the mushroom database, there are not any contradictory instances and duplicate instances. The threshold value for the noise rate is 0, because we don't want to acquire a rule with noise rate greater than 0 to discern edible and poisonous.

In the following, we describe the results for poisonous and edible respectively. The number in column *inst No.* is the number of instances covered by the rule.

### - Poisonous

The mushrooms satisfied the conditions in Table 4 are poisonous. In condition columns, the generalizations with *or* are selectable for covering the same instances. For example, we can select

$$\{A_4(f)A_7(c)A_{11}(b)A_{18}(o)\} \text{ or } \{A_7(c)A_{10}(e)A_{11}(b)A_{18}(o)\}$$

to cover instances No.49, 99, 263, 629, 6684, 6750, ..., 7899, 7940, 7995.

### - Edible

The mushrooms satisfied the conditions in Table 5 are edible. In condition columns, the generalizations with *or* are selectable for covering the same instances. For example, we can select

$$\{A_4(t)A_{22}(d)\} \text{ or } \{A_{10}(t)A_{22}(d)A_{11}(b)\} \text{ or } \{A_{19}(p)A_{22}(d)A_{10}(t)\}$$

to cover instances No.4030, 4036, 4039, 4071, ..., 5886, 5937, 6018, 6019.

In Table 4 and Table 5, we present the results for poisonous and edible respectively. We prefer the rules that are supported by as many instances as possible. If there are two or more rules supported by same instances, we prefer the rules with the most strength. For example, the following two rules have been got during our process:

$$\begin{aligned} r1. & \quad \{A_8(n)A_{15}(w)A_{20}(w)\} \rightarrow \text{poisonous} \\ r2. & \quad \{A_{15}(w)A_{18}(o)A_{20}(w)\} \rightarrow \text{poisonous} \end{aligned}$$

both of them supported by 872 instances (No.1108, 1127, 1129, 1238, 1282, 1509, 1718, 365). The strengths of them are:

$$S(\{A_8(n)A_{15}(w)A_{20}(w)\} \rightarrow \text{poisonous}) = \frac{872}{\sum_{i \neq 8, 15, 20} N_i} = \frac{872 \times N_{A_8}}{\sum_{i \neq 15, 20} N_{A_i}}$$

$$S(\{A_{15}(w)A_{18}(o)A_{20}(w)\} \rightarrow \text{poisonous}) = \frac{872}{\sum_{i \neq 18, 15, 20} N_i} = \frac{872 \times N_{A_{18}}}{\sum_{i \neq 15, 20} N_{A_i}}$$

Because  $N_{A_8} = 2$ ,  $N_{A_{18}} = 3$ ,  $S(r2) > S(r1)$ , we select *r2*.

Table 4. The conditions for poisonous

| Conditions                                                                          | Inst No. | Description                                                                                                                                 |
|-------------------------------------------------------------------------------------|----------|---------------------------------------------------------------------------------------------------------------------------------------------|
| $A_7(c)A_{12}(k)$                                                                   | (2228)   | gill-spacing=close, stalk-surface-above-ring=silky                                                                                          |
| $A_2(s)A_4(f)A_8(n)$                                                                | (960)    | cap-surface=smooth, bruises=no, gill-size=narrow                                                                                            |
| $A_5(f)$                                                                            | (2160)   | odor=foul                                                                                                                                   |
| $A_9(g)A_{11}(b)$                                                                   | (504)    | gill-color=gray, stalk-root=bulbous                                                                                                         |
| $A_{11}(b)A_{22}(g)$                                                                | (612)    | stalk-root=bulbous, habitat=grasses                                                                                                         |
| $A_{20}(r)$                                                                         | (72)     | spore-print-color=green                                                                                                                     |
| $A_3(y)A_4(f)$                                                                      | (672)    | cap-color=yellow, bruises=no                                                                                                                |
| $A_2(s)A_7(c)A_8(n)$                                                                | (1040)   | cap-surface=smooth, gill-spacing=close, gill-size=narrow,                                                                                   |
| $A_2(y)A_7(c)A_8(n)A_{13}(s)$                                                       | (560)    | cap-surface=scaly, gill-spacing=close, gill-size=narrow, stalk-surface-below-ring=smooth                                                    |
| $A_{15}(w)A_{18}(o)A_{20}(w)$                                                       | (872)    | stalk-color-below-ring=white, ring-number=one,                                                                                              |
| $A_4(f)A_7(c)A_{11}(b)A_{18}(o)$<br>or<br>$A_7(c)A_{10}(e)A_{11}(b)A_{18}(o)$       | (1392)   | spore-print-color=white<br>bruises=no, gill-spacing=close, stalk-root(bulbous=b), ring-number=one                                           |
| $A_4(f)A_{18}(o)A_{11}(b)A_{22}(d)$<br>or<br>$A_{10}(e)A_{11}(b)A_{18}(o)A_{22}(d)$ | (624)    | bruises=no, stalk-root=bulbous, ring-number=one, habitat=woods<br>stalk-shape=enlarging, stalk-root=bulbous, ring-number=one, habitat=woods |
| $A_3(g)A_4(f)A_{11}(b)$<br>or<br>$A_3(g)A_{10}(e)A_{11}(b)A_{18}(o)$                | (712)    | cap-color=gray, bruises=no, stalk-root=bulbous<br>cap-color=gray, stalk-shape=enlarging, stalk-root=bulbous, ring-number=one                |
| $A_7(c)A_{13}(k)$<br>or<br>$A_{13}(k)A_{18}(o)$                                     | (2160)   | gill-spacing=close, stalk-surface-below-ring=silky<br>stalk-surface-below-ring=silky, ring-number=one                                       |

## 7 Conclusions

In this paper, we presented a new approach based on Generalization Distribution Table (GDT) and the rough set methodology for mining *if-then* rules from databases. We described basic concepts and an implementation of our methodology. Main features of our approach can be summarized as follows:

- It can learn *if-then* rules from very large, complex databases in an incremental, bottom-up mode;
- It can predict unseen instances and represent explicitly the uncertainty of a rule including the prediction of possible instances in the strength of the rule;
- It can effectively handle noisy data, missing data and data change;
- It can flexibly select biases for search control;

**Table 5.** The conditions for edible

| Conditions                                               | Inst No. | Description                                                                        |
|----------------------------------------------------------|----------|------------------------------------------------------------------------------------|
| $A_5(n)A_8(b)A_{18}(o)$                                  | (2688)   | odor=none, gill-size=broad, ring-number=one                                        |
| $A_5(n)A_7(c)A_{12}(s)A_{18}(o)$                         | (2064)   | odor=none, gill-spacing=close,<br>stalk-surface-above-ring=smooth, ring-number=one |
| $A_8(b)A_{19}(e)$                                        | (960)    | gill-size=broad, ring-type=evanescent                                              |
| $A_1(x)A_5(n)A_8(b)$                                     | (1476)   | cap-shape=convex, odor=none, gill-size=broad                                       |
| $A_2(f)A_5(n)A_{12}(s)$                                  | (1236)   | cap-surface=fibrous, odor=none,<br>stalk-surface-above-ring=smooth                 |
| $A_3(n)A_5(n)A_8(b)$                                     | (1096)   | cap-color=brown, odor=none, gill-size=broad                                        |
| $A_1(x)A_5(n)A_{12}(s)$                                  | (1296)   | cap-shape=convex, odor=none,<br>stalk-surface-above-ring=smooth                    |
| $A_2(f)A_5(n)A_{13}(s)$                                  | (1236)   | cap-surface=fibrous, odor=none,<br>stalk-surface-below-ring=smooth                 |
| $A_1(x)A_5(n)A_{13}(s)$                                  | (1272)   | cap-shape=convex, odor=none,<br>stalk-surface-below-ring=smooth                    |
| $A_2(f)A_5(n)A_8(b)$                                     | (1392)   | cap-surface=fibrous, odor=none, gill-size=broad                                    |
| $A_4(f)A_5(n)A_8(b)$                                     | (1264)   | bruises=no, odor=none, gill-size=broad                                             |
| $A_3(g)A_5(n)$                                           | (1032)   | cap-color=gray, odor=none                                                          |
| $A_{12}(s)A_{21}(y)$                                     | (1048)   | stalk-surface-above-ring=smooth, population=solitary                               |
| $A_8(b)A_{20}(n)$                                        | (1648)   | gill-size=broad, spore-print-color=brown                                           |
| $A_8(b)A_{20}(k)$                                        | (1600)   | gill-size=broad, spore-print-color=black                                           |
| $A_4(f)A_5(n)A_7(w)A_{14}(w)$                            | (1104)   | bruises=no, odor=none, gill-spacing=crowded<br>stalk-color-above-ring=white        |
| $A_5(n)A_8(b)A_{20}(w)$<br>or<br>$A_{18}(t)A_{20}(w)$    | (528)    | odor=none, gill-size=broad,<br>spore-print-color=white                             |
|                                                          | (528)    | gill-spacing=two, spore-print-color=white                                          |
| $A_{10}(t)A_{19}(p)A_{22}(d)$<br>or<br>$A_4(t)A_{22}(d)$ | (1824)   | stalk-root=bulbous, ring-type=pendant,<br>habitat=woods                            |
|                                                          | (1824)   | bruises=bruises, habitat=woods                                                     |

- It can use background knowledge as a bias for controlling the creation of a GDT and the discovery process.

Some issues on real-world applications have not yet been solved in our approach. Discretization of continuous attributes is one among them. Like many algorithms developed in the machine learning community, our approach focuses on learning in nominal attribute space. However, since many real-world classification tasks exist that involve continuous attributes, discretization of continuous attributes must be used as a step of pre-processing in our approach. Recently, discretization of continuous attributes has received significant attention and some researchers have investigated about this [10, 1]. These results can be combined with our approach as a step of pre-processing.

## References

1. J. Dougherty, R. Kohavi, and M. Sahami. Supervised and Unsupervised Discretization of Continuous Features. *Proc. 12th Inter. Conf. on Machine Learning* (1995) 194-202.
2. Fayyad, U.M., Piatetsky-Shapiro, G., and Smyth, P. 1996. From Data Mining to Knowledge Discovery: an Overview. In *Advances in Knowledge Discovery and Data Mining*, MIT Press (1996) 1-36..
3. H. Hirsh. Generalizing Version Spaces, *Machine Learning*, Vol.17 (1994) 5-46.
4. T. Mollestad and A. Skowron. "A Rough Set Framework for Data Mining of Propositional Default Rules", in Z.W. Ras and M. Michalewicz (eds.) *Proc. Ninth International Symposium on Methodologies for Intelligent Systems (ISMIS-96)*, LNAI 1079, Springer (1996) 448-457.
5. T.M. Mitchell. Version Spaces: A Candidate Elimination Approach to Rule Learning. *Proc. 5th Int. Joint Conf. Artificial Intelligence* (1977) 305-310.
6. T.M. Mitchell. Generalization as Search. *Artificial Intelligence*, Vol.18 (1982) 203-226.
7. A. Skowron and C. Rauszer. The discernibility matrices and functions in information systems, R. Slowinski (ed.) *Intelligent Decision Support* (1992) 331-362.
8. A. Skowron and L. Polkowski, Synthesis of Decision Systems from Data Tables, in T.Y. Lin and N. Cercone (eds.), *Rough Sets and Data Mining: Analysis of Imprecise Data*, Kluwer (1997) 259-299.
9. J. Teghem and J. Charlet, "Use of Rough Sets Method to Draw Premonitory Factors for earthquakes by Emphasizing Gas Geochemistry: The Case of a Low Seismic Activity Context in Belgium", in R. Slowinski (ed.) *Intelligent decision support: Handbook of applications and advances of rough set theory*, Kluwer (1992) 165-179.
10. T.Y. Lin, *Neighborhood Systems - A Qualitative Theory for Fuzzy and Rough Sets*, in Paul Wang (ed.) Advances in Machine Intelligence and Soft Computing, Volume IV (1996) 132-155.
11. T.Y. Lin and N. Cercone (ed.) *Rough Sets and Data Mining: Analysis of Imprecise Data*, Kluwer Academic Publishers (1997)
12. Z. Pawlak. *ROUGH SETS, Theoretical Aspects of Reasoning about Data*, Kluwer Academic Publishers (1991).
13. N. Zhong and S. Ohsuga, Using Generalization Distribution Tables as a Hypotheses Search Space for Generalization. *Proc. 4th International Workshop on Rough Sets, Fuzzy Sets, and Machine Discovery (RSFD-96)* (1996) 396-403.
14. N. Zhong, S. Fujitsu, and S. Ohsuga, Generalization Based on the Connectionist Networks Representation of a Generalization Distribution Table *Proc. First Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD-97)*, World Scientific (1997) 183-197.
15. N. Zhong, J.Z. Dong, and S. Ohsuga, "Discovering Rules in the Environment with Noise and Incompleteness", *Proc. 10th International Florida AI Research Symposium (FLAIRS-97)* edited in the *Special Track on Uncertainty in AI* (1997) 186-191.

# Constructing Personalized Information Agents

Chia-Hui Chang and Ching-Chi Hsu

Department of Computer Science and Information Engineering  
National Taiwan University, Taipei, Taiwan 106

**Abstract.** The explosion of the World Wide Web as a global information network brings with it a number of related challenges for information agents. In this poster, we propose our architecture for constructing a personal information agent, which consists an web searching assistant and an interest domain manager responsible for the construction of the search domains for the user. Under the infrastructure, other information agents could be collaborated into this environment to discovery new information.

## 1 Introduction

A lot of people have been suffered from the problem of thousands of “hit” documents returned by search engines. There are two reasons for this situation: one is the difficulty for query formulation and the other is the inherent word ambiguity in natural language. And due to the network delay in today’s internet, there is a compelling need for an automatic information discovery and filtering service. Such problems have excited the motivation for constructing a personal information search agent for the user: an online information search assistant and an offline information discovery and filtering tool.

To simplify the design and focus ourselves on the retrieving and filtering techniques, the process of information search is divided into two stages: the first for distributed resource discovery and the second for information access.

- **Distributed Resource Discovery** Distributed resource discovery is the process to locate information on a large network. For a personal information agent, there are two ways to do: one is to take the advantage of web search engines such as AltaVista or WebCrawler, that apply spider or crawler techniques to archive internet homepages. The other is to traverse the hyperlinks itself from some starting hypertexts.
- **Information Access Models** The second issue is the access models, which state the problem of how information is accessed from the collection. Two approaches are generally employed: one is the subject-based browse model; the other is the keyword-based query model. In the former case, the main problem is how information could be organized. While in the latter case, the main problem is how documents are ranked.

## 2 The Targets and Solutions

For the online search assistant, the discovery process is achieved by the multi-engine search such that the main effort focuses on how information can be accessed. As we know, the main issue in query based search tools is the short query given by the user. Thus, document ranking is not an ideal way in this scenario. And due to the ambiguity of words in natural language, There must be some ways to give suggestions to the user and have them feedback to clarify the ambiguity. Thus, the best is to incorporate the browse model by appropriate topic classification. The insight is that the related information about the query topic may be embedded in the initial query results. If appropriate clustering of the data into subject/topic groups could be constructed, the ambiguity can be cleared up by exclusion of loathing documents or groups.

From the query history of the user, the search assistant is able to cache related hypertexts the user is interested in and starts the automatic discovery process. Two approaches are introduced here.

- *Hyperlink Based Traversing* The first one is site traversing agent, which traverse the hyperlinks from some starting pages cached in the query history. The idea is intuitive from the observation that hyperlinks between documents indicate the potential relationship in between. While traversing through the hyperlink chain may lead us to some collection pages and more documents could be found.
- *Query Based Discovery* The second one is an application of genetic algorithm, where each query vector is viewed as a chromosome with the words as genes. The idea is implied from the observation that different combination of words about one topic derives different search outcomes and ranking results. Hence, by combining related queries, we would be able to find out new information that has not yet been discovered.

In both cases of target-based discovery and site traversing, filtering of the new found documents remains to be the core technique. The newly found URLs are only the candidate resources to be included. The final documents suggested to the user are those with *satisfactory verification*.

## 3 Summary

In this poster, we focus ourselves on the problem of constructing personal information systems and propose a simple infrastructure towards such information systems. The designing philosophy of the online search assistant is the concept level clustering of the initial query results. Thus, textual summaries of the clusters are important in such design. Keyword extraction is the first step for textual summary, it also helps the user formulate his/her query expressions. On the other hand, with the interest domains constructed, the query-based discovery agent could regenerate query expressions using genetic algorithm and the site-traversing agent could traverse through the likely relevant documents based on its criterion.

# Towards Real Time Discovery from Distributed Information Sources

(full paper available via [www.cs.ust.hk/faculty/beat/bio.html](http://www.cs.ust.hk/faculty/beat/bio.html))

V. Cho and B. Wüthrich

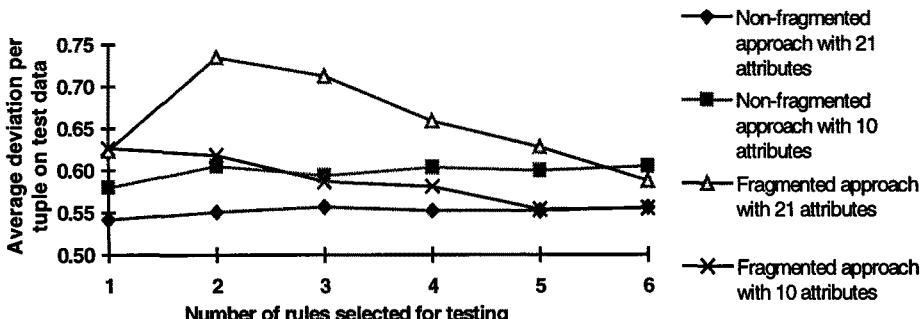
The Hong Kong University of Science and Technology

Suppose there are many data sources (e.g. Web pages) or data fragments  $db_i$  containing information about objects. We present techniques to extract knowledge from these individual data sources so that this knowledge then adequately represents the entire database  $\{db_1, \dots, db_k\}$ . Obviously, there are two possibilities to achieve this. Firstly, all the data sources are brought together so as to form a large single database. It would now be possible to mine for knowledge using any existing approach. The merging of the individual sources, however, has severe drawbacks: the merging may actually not be possible and would be definitely costly; furthermore, mining the entire database may be too time consuming. The second possibility would be to mine only one data source  $db_i$  and then assume that the found knowledge is also applicable to all the other data sources. This however could bias the result. The found knowledge then represents one data source and there is no guarantee that in the other sources the data has similar characteristics. We therefore suggest a third way of mining the knowledge. From each data source only one rule is generated. Individual rules are then brought together so as to represent adequately the knowledge of the entire database.

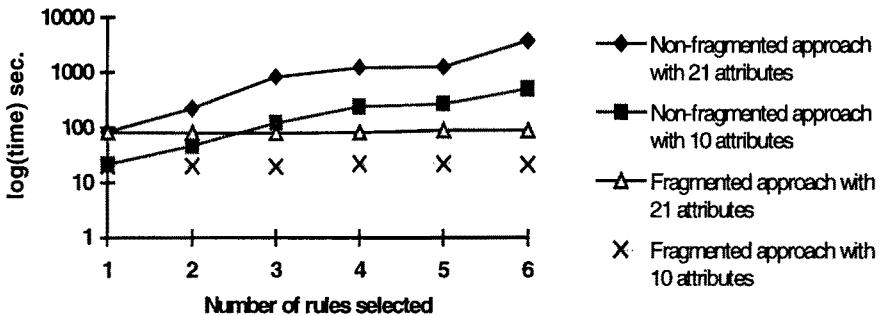
The suggested mining approach, called *fragmented approach*, works as follows. From each fragment one rule is generated. The rules generated individually are then ranked using a *ranking criterion*. The top most rules of this ranking are finally selected to form the rule set.

Classification problems are usually of categorical type. A customer is classified into either of two classes: credit-worthy (*good\_credit*) or non-credit-worthy (*bad\_credit*), but never both. We also investigate different methods to produce a consistent prediction if *good\_credit* and *bad\_credit* contradict each other such as *good\_credit* and *bad\_credit* are both '1' or they are both '0'.

As described in the full paper, ranking criterion *confidence*, and consistency making *method B* are the appropriate choices. Therefore, the next figure shows the combined performance of *method B* using *confidence* as the selection criterion.

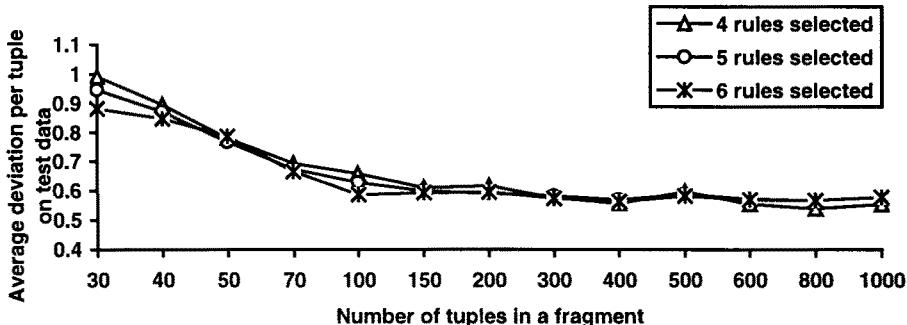


The next figure shows the logarithm of learning time against the number of rules generated for both the fragmented and non-fragmented approaches. The slope of the non-fragmented approach is approximately equal to  $\log_{10}2$ , thus the learning time doubles as the number of rules increases by one.



The average learning time of using all 21 attributes with fragmentation is 58.4 seconds and that of 10 attributes with fragmentation is 13.08 seconds on a SPARC 2 Workstation. This compares favourably with the efficiencies of the top six data mining techniques recorded for the same application. Their training time on 900 tuples ranges from 50 seconds for discriminant regression analysis to 9123 seconds for a neural network technique. The result using 21 attributes is quite close to the second best result recorded on the web site, which is 56 seconds using logarithmic discriminant regression analysis. With respect to the cost efficiency, the fragmented approach is among the best performing ones. Moreover, we find that the average time of learning from 10000 tuples is 166.7 seconds and 662.3 seconds when using 10 and 21 attributes respectively. This indicates that our rule based discovery can be done within reasonable time even for large databases.

We also investigate whether the deviation depends on the number of fragments and their sizes. The next figure shows the deviation results using different fragment sizes on a 10000 tuple database. Once the fragment size is large enough (about 100 tuples), the results are close when selecting 4, 5 or 6 rules respectively.



The presented mining approach has several advantages. First, the mining can be done much faster. Second, there is no need to physically merge the individual data sources.

# Constructing Conceptual Scales in Formal Concept Analysis

Richard Cole, Peter Eklund, and Don Walker

Knowledge, Visualisation and Order Laboratory  
Department of Computer Science, The University of Adelaide  
Adelaide 5005, Australia

**Abstract.** Formal concept analysis (FCA) [2] is a technique for knowledge visualisation. Several software tools have been developed to aid knowledge exploration using FCA, the most prevalent of these being TOSCANA[1]. This paper describes a tool to allow the creation of conceptual scales by a domain expert. The tool employs techniques of direct manipulation to provide the user with an intuitive understanding of FCA. Examples from a case study employing Medical Subject Headings (MeSH) and a set of medical texts.

## 1 Introduction

FCA aids visualisation of the dependancies between attributes possessed by a set of objects by displaying a lattice of concepts. A concept is understood to have both an extent, the set of all objects that conform to that concept, and an intent, the set of all attributes possessed by objects in the extent.

The technique of conceptual scaling in formal concept analysis allows the user to both focus her attention on a select number of attributes, and deal with multi-valued and continuous attributes.

Previous software tools supporting the application of FCA fall into two groups. Those that attempt employ conceptual scaling to navigate in the space of attributes and those that rely on navigation inside a lattice produced by all attributes and all objects.

TOSCANA/ANACONDA[1] is a software suite for the creation and combination of conceptual scales. It separates the construction and combination of conceptual scales. The reason for this is that in many applications of FCA, particularly those involving continuous many valued attributes, the construction of the conceptual scales requires extensive knowledge of the technique of formal concept analysis. One of the reasons that this extensive knowledge is required is that scales are created by entering values into a boolean array and associating rows in this array with SQL statements.

The software presented in this paper combines the construction and analysis of conceptual scales in a dynamic framework. The user is guided in the construction of conceptual scales by a direct manipulation interface.

The direct manipulation interface replies on the presence of an external hierarchy over the concepts, such as is provided over medical terms by MeSH.

## 2 MeSH Diagram

In order to construct a conceptual scale the user selects attributes by entering text searches. In the example shown in Figure 2 the attributes are terms from MeSH. All terms in MeSH have a number of text strings describing them, and these strings are matched to the user's query.

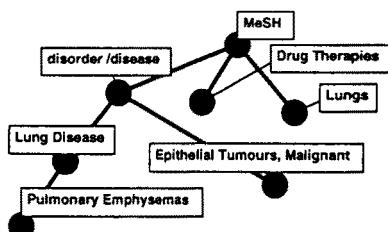


Fig. 1. MeSH Diagram

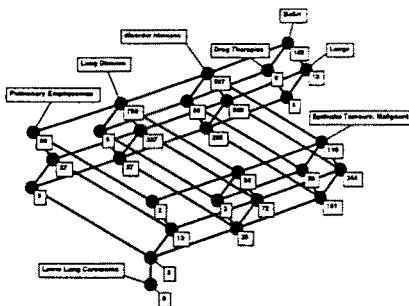


Fig. 2. Conceptual Scale

The diagram displays the generalisation hierarchy on terms from MeSH using a Hasse diagram that preserves both the join operation and the order relation from MeSH. The example shown is a tree, but not all combinations of terms from MeSH form a tree.

The MeSH diagram in Figure 1 is built incrementally as the user selects attributes. When a user selects a term from MeSH this term is added to the diagram together with the joins of the selected term and all terms currently in the diagram. An objective function is minimised to provide automated layout in which changes to the diagram are animated.

### 3 Conceptual Scales

The terms from MeSH were associated with a group of patients described by a set of medical texts. A MeSH term was associated with a patient if that term appeared in a document describing the patient. From the set of patients, this association - called an incidence relation - and the terms in the MeSH diagram created by the user, the concept lattice is created.

Figure 2 shows the concept lattice corresponding to the document set and the MeSH diagram displayed in Figure 1.

Concepts, represented by circles, have two labels attached. The numeric label is the number of patients introduced by that concept, while the text label gives the MeSH terms introduced. The intent of a concept may be retrieved by collecting attributes following lines upwards in the diagram, while the size of the concept extent is computed by summing the numeric labels going downwards.

The lattice displays implications between the attributes. If one attribute appears below another in the diagram, with a path between the two attribute concepts, then the higher concept is implied by the lower concept. Conditional probabilities may be determined by considering the size of the concept's extents.

The work is intended to find use in epidemiological studies being conducted in the Medical School at the University of Adelaide.

### References

1. F. Vogt, C. Wachter, and R. Wille. Data analysis based on a conceptual file. *Classification, data analysis and knowledge organisation*, pages 131–140. 1991.
2. R. Wille. Concept lattices and conceptual knowledge systems. *Computers Math. Appl.*, 23(6-9):493–515, 1992.

# The Hunter and the Hunted - Modelling the Relationship Between Web Pages and Search Engines

David L. Dowe, Lloyd Allison and Glen Pringle

School of Computer Science and Software Engineering,

Monash University, Clayton, Vic. 3168, Australia

{dld, lloyd, pringle}@cs.monash.edu.au

## Abstract

The major objective of a World Wide Web (WWW) search engine is to uncover WWW pages relevant to a search. The original purpose of a page might have been to inform but since the advent of search engines there has been another imperative to catch the engines' attention and to be rated as highly as possible. In turn the search engines will have to be more clever to filter out gratuitous promotion. This race is an ongoing evolutionary process.

## 1 Introduction

We gathered pages concerning 50 keywords for each of the search engines AltaVista, Excite and Lycos. These keywords were selected[1] by sampling random single word searches at Magellan's Voyeur[3] web site. For each search engine, for each of 50 randomly selected keywords[1], the top 200 documents were recorded as positive examples, and 200 negative examples were selected at random from the lists of documents for the other keywords.

The minimum message length (MML) decision tree program[4] was used to infer decision trees "explaining" the preferences of the search engines for the web pages that they returned in terms of various attributes of the pages. A decision tree is a graphical representation of a decision process.

The document attributes we considered were the number of times the word occurs in the URL, the title, the meta tag, the first `<h?>` tag and the document, the number of words in the document title, the number of words in the first `<h?>` tag and the length of the document divided by 100, rounded to nearest integer.

Page authors have long been attempting to have their pages rated more highly than others for queries related to their pages. The most common of the methods for this, word spamming, involves putting a word into the document multiple times. Most search engines penalise pages which they detect using this practice.

A decision tree[4] is a graphical way of representing a decision process. It contains leaf nodes and internal "split" nodes. A split-node specifies a test on an attribute of an item, here of a web page. Attributes that are used in split-nodes nearest to the root of the tree are, in a sense, the most important attributes. A leaf node accumulates statistics on the items that filter through the split-nodes to that leaf; here it contains the frequencies with which a search engine does or does not return such web pages. We are not suggesting that any search engine contains an actual decision tree in its inner workings but rather that decision trees can approximate many kinds of decision processes and that they are easily understood.

## 2 Results : Lycos and other engines

The decision tree runs led to several interesting results[1], which are presented as probabilistic inferences. We give some examples for Lycos[2]. The figure shows the best decision tree found at a cost of 7447.3 bits.

The first split attribute is total matches in the document. The left subtree corresponds to zero matches (<0.5) and represents documents not containing the keyword; the program finds some structure amongst them[1] corresponding to zero-length pages, "page not found", "server unavailable" and "this page has moved" etc.

Pages in the right subtree have at least one match. The next split is on document length. Short pages (<14K) have higher odds, 4543:34, of being returned than long pages, 538:113. Note that these odds refer to this particular sample and that in the whole of the world wide web there are many more negative examples, however one might reasonably expect a similar relative ordering in larger samples. It is unlikely that Lycos is explicitly targeting short pages. Rather, it is probably normalising word frequencies in some way which may inadvertently penalise long pages. The effect may also be due to the way that authors use words in short and long documents.

The leaf node described by the path right-left-right (RLR) represents short pages with a match in the first heading  $\langle h? \rangle$ . Such documents are returned in a remarkably high ratio, 1689:0. This should not be taken as a prediction of absolute certainty for all samples but it is suggestive.

The subtree right-right (RR) indicates that Lycos takes note of Meta sections. It seems to be better (77:11) to have a match in the Meta-tag than not (461:102). However, if there is an excessive number of matches in the Meta (>79, subtree RRR) then the odds are lower; the sample is small but suggests the presence of anti-spamming measures. If there are no matches in the Meta (subtree RRL) then a very high number of matches in a document (>171) reduces the odds of its being returned from 394:7 to 67:95. However a large number of matches is excused if the document is also long (>17K), the odds for long pages with many matches being 50:1 against 17:94 for shorter pages with many matches. This seems to indicate further anti-spamming measures. We do not claim that Lycos contains actual tests on the specific numbers appearing in the split-nodes of the decision tree (e.g. 79 Meta matches, 171 matches, 17K, etc.) but it seems to penalise excessive matches in some way.

## 3 Conclusion

MML decision trees were inferred to approximate and explain why search engines select the pages they return as being relevant to queries. They show that, generally speaking, "more matches" means "more relevant". However, some web pages "spam" and the engines are fighting back. We expect this battle to become more subtle.

Short documents are favoured by search engines over long documents. They are probably normalising word frequencies for document length, or perhaps weighting the start of documents more heavily. Search engines might be improved here.

Excite seems to pay the most attention to the structure of an HTML document, valuing relevant titles and headings in particular. It definitely ignores Meta information - which can easily be used for spamming. There is a speed-accuracy trade-off in the design of a search engine. The decision trees suggest that AltaVista has gone for speed and Excite for accuracy[1], confirming subjective impressions.

## References

- [1] G. Pringle, L. Allison & D.L. Dowe. *What is a tall poppy among web pages?* Accepted for Proc. of the 7th Int. WWW Conf. Brisbane, Australia, 1998.
- [2] Lycos. <URL:<http://www.lycos.com/>>.
- [3] Magellan voyeur. <URL:<http://voyeur.mckinley.com/cgi-bin/voyeur.cgi>>.
- [4] C.S.Wallace & J.Patrick. *Coding decision trees*. Machine Learning, 11:7-22, 1993

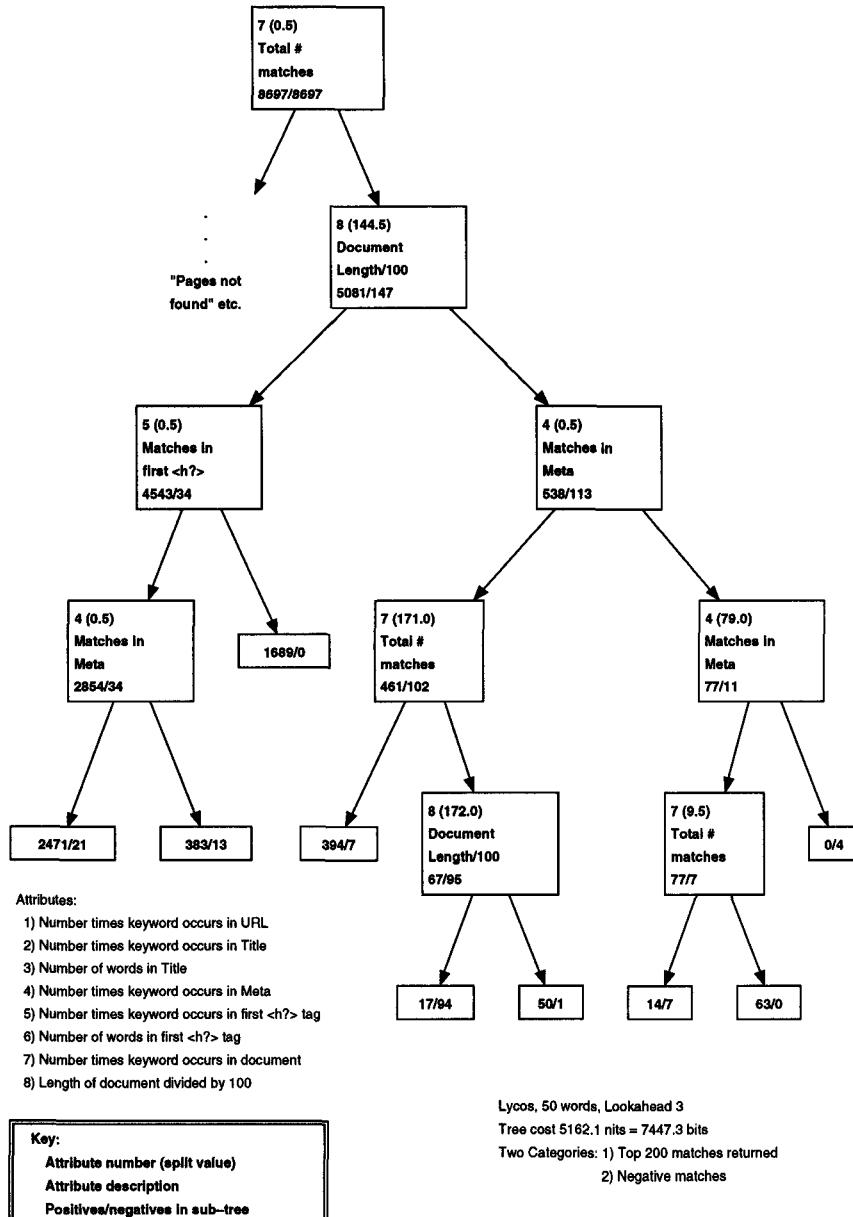


Fig. 1. Decision Tree for Lycos

# An Efficient Global Discretization Method

K. M. Ho and P. D. Scott

Department of Computer Science, University of Essex, UK  
`hokokx@essex.ac.uk` and `scotp@essex.ac.uk`

## 1 Introduction

In a previous publication [2], we described a new technique for discretization of continuous variables based on zeta, a measure of strength of association between the continuous variable and the classification variable to be predicted, that we have developed for this purpose. Experimental evaluations have demonstrated that the zeta method is both effective and fast. However, the techniques presented in our earlier paper were limited in that the number of partitions produced in the continuous variable was always equal to the number of categories in the classification variable. This paper describes a generalization of the zeta method that removes this restriction.

## 2 An Improved Version of Zeta

Suppose we have a continuous variable A and a class variable B where A is used to predict the values of B. Given a sample of  $N$  items whose value distribution is given in a  $k$  by  $j$  table where  $k$  is the number of partitions in A and  $j$  is the number of classification values in B. The generalized form of zeta is

$$Z = \frac{\sum_{i=1}^k n_{f(1),j}}{N} \times 100\% \quad (1)$$

where  $n_{ij}$  is the count in  $A_i B_j$  and  $f(i)$  should be understood as follows. In order to make predictions, each of the  $k$  partitions of A must be paired with any value of B subject to two constraints: a) at least two distinct classification values are predicted, and b) no two adjacent partitions of A predict the same value of B. If A has  $k$  distinct partitions and B has  $j$  distinct values, there will be close to but less than  $j^k$  ways in which such sets of pairings could be made. One such set of pairing will give the greatest predictive accuracy; call this the best pairing assignment. Then  $A_{f(i)}$  is the partition of A that is paired with  $B_j$  in the best pairing assignment.

## 3 Empirical Evaluation

We compared the improved zeta discretization procedure with the old zeta discretization procedure [2], entropy [1] and C4.5 [5] on twenty different datasets selected from UCI repository. C4.5 uses a local discretization technique whereas the other techniques perform discretization globally. The experiments were run using the MLC++ [4]. The results are shown in our technical report [3].

There is no one method that consistently produces better classification accuracies in all the datasets. The classification accuracies of the new zeta method improve slightly, but not significantly, over the old zeta method. However, in our experiment the local method performs only slightly, but not significantly, better

| Datasets | C4.5  | Entropy | ZetaOld | ZetaNew |
|----------|-------|---------|---------|---------|
| Satimage | 12:17 | 1:21:57 | 43:24   | 9:36    |
| Segment  | 3:48  | 15:53   | 8:41    | 3:15    |
| Shuttle  | 7:53  | 49:01   | 45:06   | 5:04    |

**Table 1.** Computation time in mm:ss with 5-folds cross-validation.

than the global method. When C4.5 uses its own local discretization technique, it produces deeper decision trees than the global methods. This suggests that the local method would produce more complicated rules from the decision trees.

The new zeta method runs consistently faster than all the other methods. Table 1 shows the computation time required by each method on selected datasets. The improved zeta method requires markedly less computation time compared to the old zeta method and entropy method. Thus, the improved zeta technique is superior to other methods because it produces similar accuracies and requires less time.

One of the reasons for developing the generalized zeta method was to allow the continuous variables to be partitioned into any number of sub-ranges thus enabling it to reflect bivariate relationship more accurately. We were therefore surprised to find that it did not lead to better predictive accuracy. One possible explanation is overfitting when the continuous variables are partitioned into sub-ranges that predict accurately the training sets but not the test sets. Further work is being carried out to investigate this problem.

These results show that zeta discretization method is both an effective and a computationally efficient method of partitioning continuous variables for use in decision trees. From the experiments it would appear that zeta discretization is the method of choice.

#### 4 Acknowledgements

We are grateful to the ESRC's programme on the ALCD for supporting part of the work reported in this paper under grant number H519255030.

#### References

1. Fayyad, U.M., & Irani, K.B. (1993). Multi-interval discretization of continuous-value attributes for classification learning. In Proc. 13th International Joint Conference on Artificial Intelligence, pp. 1022-1027. San Francisco: Morgan Kaufmann.
2. Ho, K.M., & Scott, P.D. (1997a). Zeta: A global method for discretization of continuous variables. In KDD97: 3rd International Conference of Knowledge Discovery and Data Mining, pp. 191-194, Newport Beach, CA.
3. Ho, K.M., & Scott, P.D. (1997b). An Efficient Global Discretization Method. Technical Report CSM-296, Department of Computer Science, University of Essex, England.
4. Kohavi, R., John, G., Long, R., Manley, D. & Pfleger, K. (1994), MLC++: A machine learning library in C++. In Tools with Artificial Intelligence, IEEE Computer Society Press, pp. 740-743.
5. Quinlan, J.R. (1996). Improved use of continuous attributes in C4.5. Journal of Artificial Intelligence Research 4 pp. 77-90.

# Learning User Preferences on the WEB

François Jacquet and Patrice Brenot

LIRISIA, Faculté des Sciences Mirande, BP 400, F-21011 Dijon, France  
e-mail : {Jacquet, Brenot}@crid.u-bourgogne.fr

**Abstract.** We present a new tool called INDWEB, based on Inductive Logic Programming, that can learn some concepts that characterized interesting pages for a user or a group of users with respect to a set of criteria on these pages but also on these users or group of users.

**Keywords :** Inductive Logic Programming, WEB, Information Retrieval.

## 1 Introduction

Discovering some information on the web is not always an easy task. Even the use of a search engine may not be a good help. To compensate for this problem some systems have been designed to integrate some techniques from artificial intelligence. WebWatcher [2], Letizia [4], Lira [1], Syskill&Webert [7], Webdogie [3] are some of such systems. What must be noted is that all these systems focus on the research of a set of keywords in the WEB pages browsed by the user. Learning, in these systems mainly consists in observing the words that occur in the pages downloaded and comparing those words with the words found in the previously downloaded pages which were pointed out by the user as some interesting pages.

We developed a prototype, called INDWEB, that can help Internet users to browse the WEB by learning a model of their preferences. INDWEB is based on Inductive Logic Programming [6], a machine learning technique that learns Logic Programs, modelling a concept, from a set of observations on that concept.

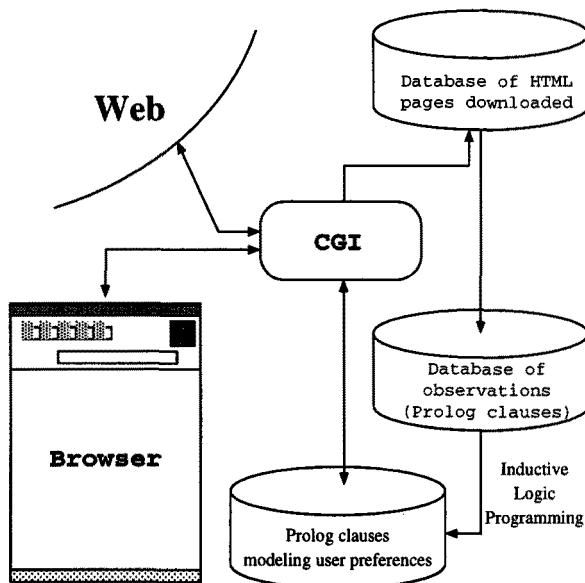
## 2 Inductive Logic Programming for Intelligent WEB Browsing

The main architecture of INDWEB is given by Figure 1.

When a user wants to use INDWEB, he has to connect using the URL of the system (non-public URL at the moment). Then he has to identify himself using a login name and a password. Then he can connect his machine to a WEB site by giving the URL of the site in a form designed in that matter in the home page of INDWEB. From that time, all the requests pass through a CGI script which allows a connection to the WEB server asked for, the download of the HTML page and several processes on this page:

- Parsing of the page according to some criteria : logical name, URL, network domain, keywords contained, writing language, number of pictures, complexity, average number of visits per day, users who already visit this page...

- Storage of information extracted from the page.
- Use of the Prolog program that models the users preferences to insert, in the page, informations that can put some links and the general interest of the page forward. This Prolog program is generated by the ILP module of INDWEB.
- Display of the enhanced page.



**Fig. 1.** Architecture of the INDWEB system

The kernel of the machine learning component is based on an ILP system. Both Progol [5] and CLAUDIEN [8] have been tested in a first version of INDWEB.

The background knowledge of this ILP component is made up of information about the users (age, education level, languages mastered, ...) and groups of users (collaborative learning).

### 3 Comparison with related works

The use of ILP in INDWEB is something very important compared with all the techniques integrated in the other systems. Indeed, due to the fact that the underlying formalism of ILP is logic programming, users preferences may be expressed in terms of features about that user or the page he is browsing but also in terms of relations between features of other pages and of other users.

Moreover, INDWEB offers the advantage of considering not only the keywords of the pages being browsed, what is the main task of the other systems. INDWEB can learn users preferences in term of many other important features such as the esthetic of this page, the number of visit, etc...

The set of the various features (of pages, users, groups of users) used by INDWEB may be easily extended. It is possible to add some other features in a form designed for that task. That leads later to the generation of other Prolog facts and eventually later to the induction of new Prolog rules. These extensions are made by an administrator of the INDWEB system on a site and not directly by the final user.

## 4 Conclusion and Future works

The power of INDWEB relies on the formalism it uses. With logic programming, concepts learned by INDWEB can be defined in terms of relations between various features of pages, users or group of users.

In the framework of future developments, we want to introduce a score for each page downloaded by a user. In that way, a page would no more be only interesting or not but, rather, it could be possible to reflect a certain level of interest for a given user.

Features associated with pages, users and groups of users are currently being refined. Concerning keywords for instance, it is obvious that syntactic and semantic contexts may influence the score of a page.

## References

1. M. Balabanovic and Y. Shoham. Learning information retrieval agents : Experiments with automated web browsing. In *AAAI Spring Symposium Series on Information Gathering from Distributed, Heterogeneous Environments, Working Notes*, AAAI Press, 1995.
2. T . Joachims, D. Freitag, and T. Mitchell. Webwatcher : A tour guide for the world wide web. Technical report, Carnegie Mellon University, September 1996.
3. Y. Laskari. The webhound personalized document filtering system. <http://www.media.mit.edu/projects/webhound/doc>. 1994.
4. H. Lieberman. Letizia : An agent that assists web browsing. In *Proceedings of the IJCAI'95*, Montréal, August 1995.
5. S. Muggleton. Inverse entailment and progl. *New Generation Computing*, 13(3-4):245–286, 1995.
6. S. Muggleton and L. De Raedt. Inductive logic programming : Theory and methods. *Journal of Logic Programming*, 19-20:629–679, 1994.
7. M. Pazzani, J. Muramatsu, and D. Billsus. Syskill & webert : Identifying interesting web sites. In *Proceedings of AAAI Conference*, Portland, August 1996.
8. L. De Raedt and L. Dehaspe. Clausal Discovery. *Machine Learning*, 26:99–146, 1997.

# **Using Rough Sets for Knowledge Discovery in the Development of a Decision Support System for Issuing Smog Alerts**

Ilona Jagielska

School of Information Management & Systems, Monash University,  
PO Box 197, Caulfield East, Victoria 3145 Australia  
Email: Ilona.Jagielska@sims.monash.edu.au

## **1. Introduction**

Real-life databases often contain data that is ambiguous, incomplete, and noisy, which makes it difficult for many of these techniques to uncover patterns in the data. There is a need, therefore, for knowledge discovery techniques that can identify patterns under noisy conditions. When extracted patterns are used for decision support it is particularly important that they are represented in a form understandable by decision makers.

The rough sets approach proposed by Pawlak (see Pawlak, 1991) is primarily concerned with indiscernibility and vagueness in data. It provides rigorous mathematical techniques for discovering regularities in data and is particularly useful for dealing with imprecise and inconsistent information. Rough sets have been successfully applied in the area of automated knowledge acquisition and knowledge discovery (Ziarko 1996).

In this research rough sets were applied to knowledge discovery for decision support in the area of issuing smog alerts for Melbourne. In particular, rough sets were used to discover and analyse patterns and relationships in data which was recorded and used by the EPA for smog events forecasting. The recorded data is noisy, imprecise, and often contradictory. The existing forecasting scheme is based on experts' assessment of certain meteorological conditions.

## **2. The application**

In 1984 the EPA in Melbourne developed a smog alert forecasting scheme which is based on assessment of meteorological conditions. When a smog event is likely to occur, a form which seeks answers to ten questions relating to the meteorological conditions is completed and used by a meteorologist as input to his/her decision whether to issue a smog alert for the next day. According to the EPA the current forecasting scheme is inadequate as its average accuracy has been around 40% correct forecasts. A performance level of 60 % is considered adequate, however, the desirable level of performance is 80% and above( Cook, 1995).

The EPA smog events forecast database consists of the following groups of data: set 1 - this set has ten attributes (yes/no answers) which correspond to ten questions about the meteorological conditions, set 2 - actual meteorological conditions. For this study, 259 forecast days were used.

## **3. Models and results**

The experiments were conducted using DataLogic/R software from *Reduct Systems Inc.* Users can control model generation through two parameters: roughness and rule

precision threshold. The rule roughness changes the amount of detail in the rules while the rule precision threshold is used to control the minimum rule probability. The system also generates a list of non-redundant attributes and their significance.

This paper describes only models built using the meteorologist's questions and answers about meteorological conditions (set1). The purpose of this study was to determine the importance of each question for smog event forecasting and to establish whether it is possible to predict smog events with an acceptable accuracy from the yes/no answers. The predictive capability of each generated model was tested. In the experiments the level of roughness was gradually changed from very high to low values. In each experiment a minimal set of attributes for the particular level of roughness and the relative significance of the attributes was generated. Around 50 different models were generated of which four are discussed in this paper. Selected results are shown below.

For roughness  $\leq 0.8$  the reduct contained nine attributes. Five strongest attributes and their strength were: pressure gradient (0.32), thickness change (0.22), wind direction (0.19), wind speed (0.17), weekend/holiday (0.14). The question about temperature inversion was not included in the reduct which indicates that this parameter had no importance for the prediction of smog events in this model. The system generated 10 rules for the Yes decision and 8 rules for the No decision. The overall classification rate of the model was 73.85%. The generalisation capability of the model (avg. accuracy on testing) was 63.1%.

#### **4. Discussion of the results**

The research provided an interesting insight into the nature of the problem of forecasting smog events. In particular, it found that out of all the meteorological factors recorded in the forecasting database the pressure gradient, and thickness change had the highest significance. The questions from the forecast form about Mt Gambier and Laverton winds had no importance for the prediction. Temperature inversion was found to be irrelevant to the prediction, and the questions about sunshine forecast and location of the high pressure system also had significance close to zero.

The results indicate that the patterns in the smog event forecasting database are weak and have poor predictive capability. If there are strong patterns in data, the attribute strength in each class is close to 1. Although the accuracy of the models generated in this study was better than the accuracy of the current forecasting scheme it is still unsatisfactory and the problem needs further investigation. Further research has been directed towards the inclusion in the model some of the meteorological data from set 2 and identification of other factors that are conducive to the occurrence of smog events.

#### **References**

- (Cook, 1995) B. Cook, Smog Alert Review - Phase 1 Review of the Forecasting Scheme 1984-1994, *Internal Working Document, EPA*, July 1995.
- (Pawlak, 1991) Z. Pawlak, Rough Sets: Theoretical Aspects of Reasoning About Data, *Kluwer Publishers*, 1991.
- (Ziarko, 1996) W. Ziarko., Discovering Classification Knowledge in Databases Using Rough Sets, in *The Second International Conference on Knowledge Discovery&DataMining, KDD-96*

# **Empirical Results on Data Dimensionality Reduction Using the Divided Self-Organizing Map**

Takamasa KOSHIZEN, Dept. of Systems Engineering, Australian National University,  
Canberra, ACT, 0200, Australia.

Hiroaki OGAWA, International Computer Science Institute, Berkeley, CA 94704, USA.

John FULCHER, Dept. of Computer Science, University of Wollongong  
Northfields Avenue, Wollongong, NSW, 2522, Australia

## **1. The Divided Self Organizing Feature Map (DSOM)**

In order to reduce the complexity of the classification task and decrease the quantisation error, we propose a speech data classification framework based on a two-stage neural network model comprising the *divided self organizing map (DSOM)* and multilayer perceptrons (MLP). Initially, the speech features are extracted using the RASTA-PLP (RelAtative SpecTrAl - Perceptual Linear Predictive) speech analysis technique, based on the filtering of time trajectories of outputs from critical-band filters [1]. Each DSOM then projects the N-dimensional speech features onto an M-dimensional space ( $N > M$ ). The transformed feature vectors are then classified by the MLP. The main advantage of the proposed scheme is the dimensionality reduction of the feature spaces by DSOM. In actual, DSOM has the same topology as the original SOM except that the MAP is divided into k smaller maps. Each node corresponds to the subset of the input vector dimensionality. For instance, our 32x32 original SOM map of the 17 dimension input vector is divided into 4 sets of 16x16 SOM maps, and the input vector is also divided into 4, 4, 4, and 5 dimension vectors for each maps respectively. Employing several DSOMs, results in a lower quantisation error compared with that of a single original SOM. Each set of k input nodes are then assigned to a DSOM. Accordingly, the number of the winner nodes will multiply, depending on the value of k. Now, since each winner node is a 2 dimensional coordinate ( $M=2$ ), this results in  $(2*k)$  dimensionality overall.

## **2. Results and Discussion**

The speech data we used is “30K Numbers”, a collection of spontaneous ordinal and cardinal number, continuous digit strings and isolated digit strings [2]. Each frame of the speech signal was analyzed into 9 dimensional *speech feature vectors*. The difference in short time periods is also calculated for 8 features of the feature vector. Thus, we get 17 dimensional feature vectors for each frame. The classifier used in all experiments was an MLP with one hidden layer. More precisely, it consisted of 400 hidden units that receive 153 input units (an input frame with 8 frames of acoustic context), and 56 output units.

The performance of the proposed scheme was analysed using an Quantisation Error and Classification Rate. In the first experiment, the original SOM size was set to be constant (  $32 \times 32 = 1024$  nodes ). The original map was then divided into 2, 4, and 8 smaller maps (DSOM). The second experiment, the original SOM size was reduced while the number of DSOM and data sets were set to be constant.

**Experiment 1. (SOM size is constant-32x32)**

| Number of DSOM and Data Sets | Total Input Dimensionality | SOM (Output) Dimensionality | Quantisation Error | Classification Rate |       |
|------------------------------|----------------------------|-----------------------------|--------------------|---------------------|-------|
|                              |                            |                             |                    | Training            | Test  |
| 1                            | 17                         | 2 (15)*                     | 2.64               | 54.25               | 53.72 |
| 2                            | 17                         | 4 (13)                      | 2.20               | 58.30               | 57.02 |
| 4                            | 17                         | 8 (9)                       | 1.65               | 67.04               | 65.06 |
| 8                            | 17                         | 16 (1)                      | 0.397              | 78.19               | 74.90 |

**Experiment 2. (Number of DSOM and Data Sets is constant-8)**

| Number of DSOM and Data Sets | Total Input Dimensionality | SOM Size | Quantisation Error | Classification Rate |       |
|------------------------------|----------------------------|----------|--------------------|---------------------|-------|
|                              |                            |          |                    | Training            | Test  |
| 8                            | 17 (1)*                    | 32x32    | 0.397              | 78.19               | 74.90 |
| 8                            | 17 (1)                     | 16x16    | 0.622              | 78.26               | 74.28 |
| 8                            | 17 (1)                     | 8x8      | 1.050              | 77.68               | 73.44 |

\* ( ) denotes the total dimensionality reduction

Results from the experiment 1. showed that the DSOM constrained the Quantisation Error more than the original SOM. Further experimentation led us to conclude however that the clustering method based on SOM may not be an appropriate dimension reductor capable of mapping from high-dimensional data space to a smaller-dimensional one. Firstly, there are no exact criteria for minimizing the classification error (MCE), apart from the Quantisation Error. Thus we have to choose SOM parameters by trial and error over the MCE. In other words, selection of the most suitable parameters is still an *open* problem which needs further investigation. Secondly, it is well-known in competitive learning that the larger the size of SOM, the larger is the number of the empty nodes which do not correspond to *any* features [3]. The results from our experiment also showed that the classification rate did not significantly improve, although the quantisation error decreased (Experiment.2).

Although there are a couple of weak points of the SOM-based data dimensionality reductor described above, SOMs has been exploited in many applications because of their ability to generate topology-preserving and dimensionality-reducing mappings. The further development of SOM-based dimension reductor, for the speech classification is left for a future study.

## Bibliography

1. Hermansky, H., Morgan, N., Bayya, A. and Kohn, P. (1991),"RASTA-PLP Speech Analysis", International Computer Science Institute, Berkeley, CA, Technical Report, TR-91-069.
2. CLSU (Center for Spoken Language Understanding) Corpora Homepage, 30K Numbers Database, [Http://www.cse.ogi.edu/CLSLU/corpora/corpus-list.html](http://www.cse.ogi.edu/CLSLU/corpora/corpus-list.html).
3. Ueda, N. and Nakano, R. (1995), "Competitive and Selective Learning Method for Vector Quantizer Design - Equidistortion Principle and Its Algorithm.", Systems and Computer (Japan), Vol. 26, No. 9.

# Mining Association Rules with Linguistic Cloud Models

Deyi Li\* Kaichang Di\*\* Deren Li\*\* Xuemei Shi\*\*\*

(\*Institute of China Electronic System Engineering,  
No.6, Wanshou Rd, Beijing, P. R. China , 100036)

(\*\*School of Information Engineering, Wuhan Technical University of  
Surveying and Mapping, No.39, Luoyu Rd, Wuhan, P. R. China , 430070)

(\*\*\* Dept of Computer Science, Hong Kong Polytechnic Univ. Hong Kong)

## 1. Introduction

Mining association rules is an important issue in KDD applications. Since it was introduced by Agrawal et al., various algorithms have been proposed and developed. But studies on knowledge representation and uncertainty handling in mining of association rules are seldom reported.

In this paper, we describe the linguistic cloud model and extend it to two and multi-dimensional ones and explore the application of cloud models in mining association rules at any concept level or at multiple concept levels by combining with Apriori algorithm.

## 2. Linguistic Cloud Models

### 2.1 Linguistic Cloud Models

A compatibility cloud is a mapping from the universe of discourse  $U$  to the unit interval  $[0,1]$ . It is a one-point to multi-point transition, producing a compatibility cloud, rather than a membership curve. A normal compatibility cloud is characterized with three digital parameters:  $Ex$ ,  $En$ , and  $D$  which are the expected value, entropy and deviation of the cloud respectively. The degree of membership  $u$  to linguistic term  $T$  is a probability distribution rather than a fixed value. The mathematical expected curve (MEC) of a compatibility cloud may be considered as its membership function from the fuzzy set theory point of view.

A two dimensional compatibility cloud is a mapping from the 2-D universe of discourse  $U$  to the unit interval  $[0, 1]$ . It is characterized with six parameters supposing the two dimensions of the universe of discourse are independent. We can extend the linguistic cloud model to multi-dimension one further. Suppose a universe of discourse have  $n$  dimensions which are independent from each other. The  $n$  dimensional normal compatibility cloud for a linguistic term in the universe is characterized with  $3n$  digital parameters.

### 2.2 Cloud Generators

There are two kinds of atom generators: forward and backward generators. The combination of the two kinds of generators can be used interchangeably to bridge the gap between quantitative and qualitative knowledge.

Cloud drops may be generated on conditions. Under the condition of a given numerical value  $u$  in the universe of discourse  $U$ , all the drops have the same value  $u$ , and probability distributed membership degrees; whereas under the condition of a

---

This research was supported by the research grant NNSFC 49631050 from the National Nature Science Foundation of China

given membership degree, all the drops have the same membership degree, and normal distributed numerical values in the universe of discourse.

### **3. Mining Association Rules with Linguistic Cloud Models**

#### **3.1 Attribute Generalization Based on Linguistic Clouds**

Generally, strong rules are likely to exist at high concept levels. When attributes are numerical, it is preferable to generalize the attributes first and then mine rules in the generalized data. Attributes are considered as linguistic variables. With each linguistic variable, several linguistic clouds are given to represent linguistic terms of the variable. Attribute values are assigned to linguistic terms by the principal of maximum compatibility value.

Cloud model based generalization allows the overlapping area between neighbor linguistic terms. Because compatibility values are random numbers, some attribute values within the overlapping area of two neighbor clouds may be assigned to different clouds at different occasions. Therefore it is a soft partition of attribute spaces that can mimic human being's thinking better than the conventional crisp partition method.

#### **3.2 Virtual Clouds**

We define two kinds of virtual clouds: floating clouds and synthesized clouds. Floating clouds are very useful when user specified clouds for a linguistic variable are not enough to cover the universe of discourse. A synthesized cloud is used to synthesize linguistic terms into a generalized one. The mechanisms of virtual cloud construction provide a general way for attribute generalization at multiple concept levels.

#### **3.3 Experiment and Discussion**

We combine the linguistic cloud models with Apriori algorithm to discover association rules in a spatial database about the geospatial and economic information in China mainland. The size of the task relevant data is about 600K bytes. Eight rules are discovered, which reveal the relation of average income to location, road density and distance to the sea. They are reduced to six rules at a more abstract concept level. Four rules about the relation of road density to elevation and location are also discovered. The experiment shows the benefits on effectiveness, efficiency and flexibility of the linguistic cloud model for mining of association rules.

## **4. Conclusion**

The linguistic cloud model is used to generalize attribute values for data preprocessing of mining association rules. The two dimensional and multi-dimensional linguistic cloud models are presented as the extension of the previous 1-D model. Combining the cloud model based generalization method with Apriori algorithm can mine association rules at any concept level or at multiple concept levels effectively and flexibly.

#### **References (omitted)**

# A Data Mining Approach for Query Refinement

Ye Liu<sup>1</sup>, Hanxiong Chen<sup>2</sup>, Jeffrey Xu Yu<sup>3</sup> and Nobuo Ohbo<sup>1</sup>

<sup>1</sup> Department of Electronic & Information Science,  
University of Tsukuba, Tsukuba, Japan 305.

<sup>2</sup> Tsukuba International University, Manabe 6, Tsuchiura, Japan 300.

<sup>3</sup> Department of Computer Science, Australian National University,  
Canberra, ACT 0200, Australia.

**Abstract.** We propose a data mining approach for query refinement using Association Rules (ARs) among keywords being extracted from a document database. We are concerned with two issues in this paper. First, instead of using minimum support and minimum confidence which has little effectiveness of screening documents, we use maximum support and maximum confidence. Second, to further reduce the number of rules, we introduce two co-related concepts: “stem rule” and “coverage”. The effectiveness of using ARs to screen is reported as well.

## 1 Introduction

Due to the rapid growth of on-line documents, the difficulty users are facing is that a large number of irrelevant documents will be returned when a query is under-specified or contains ambiguous keywords. However, the task of formulating an effective query is difficult in the sense that it requires users, without any knowledge about the collection of documents, to predict the keywords that will appear in the documents the users want to have. We use Association Rules (ARs), and we focus on screening retrieval results. ARs are based on a statistical method in which the support makes rules meet certain frequency and the confidence describes degree of relation. The confidence also indicates that relationships are not symmetric, as confidences of  $X \Rightarrow Y$  and  $Y \Rightarrow X$  are often different from each other.

In our scheme, first, the keywords will be extracted from a document database. ARs are mined for those keywords which have a good screening effect, and are stored in a rule base. Second, the system selects ARs from the rule base with respect to a user’s query (a set of keywords). The ARs will be displayed including the support and the confidence for each association rule. Third, the user can make a decision to refine his/her original query repeatedly.

## 2 Stem Rule and Coverage

Given a query containing a set of keywords  $Q$ . When an AR  $\{X \Rightarrow Y, \%C\}$  with a high confidence is picked up, the keywords included in  $Y$  will be added to a

submitted query  $Q$ . However, when the AR has a high confidence, the effectiveness of screening of the number of documents being retrieved is very small. To address this issue, maximum support and maximum confidence are proposed to be used in conjunction with minimum support and minimum confidence.

Let  $\mathcal{D}$  and  $\mathcal{K}$  be a set of documents and a set of keywords, respectively. An operation  $\rho$ , which extracts keywords from a document  $d \in \mathcal{D}$ , is defined as  $\rho(d) = \{k \mid (k \in \mathcal{K}) \wedge (k \text{ is a keyword included in } d)\}$ . Furthermore, let  $D \subset \mathcal{D}$ .  $\rho(D) = \bigcup_{d \in D} \rho(d)$ , and in particular  $\rho(\mathcal{D}) = \mathcal{K}$ . A query is a set of keywords  $Q (\subset \mathcal{K})$ . A query evaluation will retrieve all the documents that contain all the given keywords in  $Q$ , and is defined as  $\sigma(Q) = \{d \mid d \in \mathcal{D} \wedge Q \subseteq \rho(d)\}$ . Let  $X$  and  $Y$  be subsets of  $\mathcal{K}$ . The support and the confidence of a rule,  $X \Rightarrow Y$ , are calculated as follows:  $Spt(X \Rightarrow Y) = Pr(X \cup Y)$  and  $Cnf(X \Rightarrow Y) = Pr(X|Y) = Pr(X \cup Y)/Pr(X)$ . Here,  $Pr(K) \stackrel{\Delta}{=} |\sigma(K)|/|\mathcal{D}|$  where  $K \subset \mathcal{K}$ .

**Definition 1. Base Condition:** A rule,  $X \Rightarrow Y$ , satisfies a base condition if and only if  $\theta_{s_l} < Spt(X \Rightarrow Y) < \theta_{s_u}$ , and  $Cnf(X \Rightarrow Y) < \theta_{c_u}$ . Here,  $\theta_{s_l}$ ,  $\theta_{s_u}$  and  $\theta_{c_u}$  are minimum support, maximum support and maximum confidence of the rule, respectively.

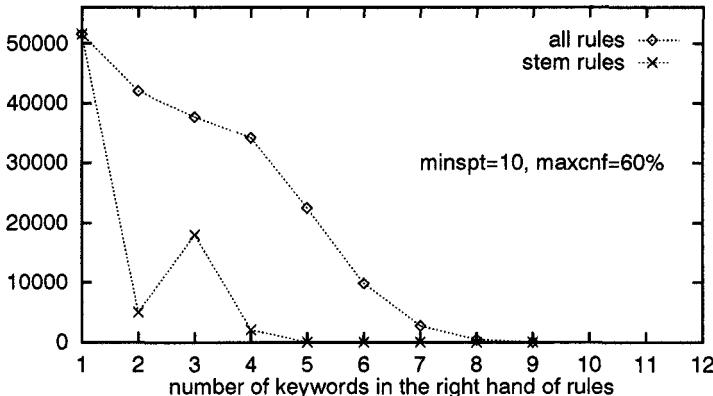
To avoid a large number of rules being generated, *stem rules* are introduced by which the other association rules can be derived. The stem rules are the only rules that we need to store in the rule base. All the other applicable association rules can be generated from the stem rules at run time instead of being generated from scratch. Furthermore, a co-related concept, called *coverage*, is proposed. A set of keywords is said to be a coverage of a set of documents if these documents can be retrieved using the same set of keywords. A minimum coverage is the smallest set of keywords to cover the documents. With the concept of stem rules and coverage, the number of ARs can be reduced to a feasible level, and it can ensure that all documents can be covered. The algorithm for generating stem rules is found in [2]. The algorithm for generating coverage is also developed.

**Definition 2. Stem Rule:** Given two rules  $r_1$  and  $r_2$ . If  $r_1$  satisfying the base condition implies that  $r_2$  satisfies the same base condition, we say that rule  $r_2$  is derived from rule  $r_1$ . A stem rule is a rule that can not be derived by any rules. In particular, for a stem rule,  $X \Rightarrow Y$ , there exists  $d \in \mathcal{D}$  such that  $X \cup Y \subset \rho(d)$ .

**Definition 3. Coverage:** Let  $K$  and  $D$  be a subset of  $\mathcal{K}$  and a subset of  $\mathcal{D}$ , respectively. We say that  $K$  covers  $D$ , or  $K$  is a coverage of  $D$ , iff  $D \subseteq \bigcup_{k \in K} \sigma(k)$ .  $K$  is called a minimum coverage of  $D$  iff no pure subset  $K'$  of  $K$  covers  $D$ .

### 3 Experimental Results

Our prototype system currently supports a medium-sized document database of electric engineering documents. The number of all documents is 40,000 ( $|\mathcal{D}| = 40K$ ), and 16717 keywords are extracted from the document database ( $|\mathcal{K}| =$



**Fig. 1.** The numbers of rules begin generated.

$\rho(\mathcal{D}) = 16717$ ). A document contains at least 3 keywords and at most 33 keywords. Two-thirds of the keywords have supports less than 10 documents. The numbers of rules we generated with  $\theta_{s_l} = 0.025\% (= 10/40,000)$ ,  $\theta_{s_u} = 5\%$  and  $\theta_{c_u} = 60\%$  are shown in Figure 1. Here, the number of rules is counted according to the number of keywords used in the rules. From Figure 1, it can be concluded that even an extremely small  $\theta_{s_l}$  can reduce the size of rule base considerably, and can be effective for screening. Using a large  $\theta_{s_l}$  can further reduce the size of the rule base, but the stem keywords might not cover all documents. From Figure 1, it can be seen that both Agrawal's algorithm[1], and our algorithm will generate the same 50000 rules when there is only one keyword. This is because that no combination of keywords is needed in both algorithms. However, for any large number of keywords, when all combinations of keywords must be checked, Agrawal's algorithm needs to generate totally about 150,000 rules while our algorithm only generates 20,000 rules.

We also submit queries and investigate how many documents the queries will get. We confirm that the average number of documents being retrieved using ARs is reduced to less than one-third.

## References

1. Agrawal R., Imielinski T., Swami A.: Mining Association Rules between Sets of Items in Large Databases. *ACM SIGMOD '93*, 207–216.
2. Chen H., Liu Y., Ohbo N.: Keyword Document Retrieval by Data Mining. *IPSJ SIG-NOTES 97(64)*(1997), 227-232. (in Japanese)

# CFMD: A Conflict-Free Multivariate Discretization Algorithm

(*Extended Abstract*)

*Ying Lu, Huan Liu, Chew Lim Tan*

National University of Singapore  
`{luy,liuh,tanc1}@iscs.nus.edu.sg`

Most existing discretization algorithms merge intervals of a continuous attribute solely base on the correspondence of the values of that particular attribute and the class labels. In most real world data, the class label is usually dependent on the combination of the values of attributes. Thus, deciding the intervals of a continuous attribute by the class labels and the values of that attribute alone may not produce the optimal intervals. This deficiency may be avoided if we merge the intervals of an attribute by taking into account the values of the other attributes as well as that particular attribute.

In this paper, we propose a randomized multivariate discretization algorithm, CFMD. The unique feature of this algorithm is as follows: When an attribute is being discretized, not only the values of the attribute to be discretized, but also the values of other attributes in the same tuple are used to determine whether two values should be merged into the same interval. The merging criterion of CFMD is whether there is a conflict. Two adjacent values are merged if there is no class conflict. Class conflict occurs when two or more tuples having exactly the same values for all the attributes but different class labels. For two tuples, if there is at least an attribute  $A_k$  such that  $s.A_k \neq t.A_k$ , then we say there is no conflict. This is because, though the class labels of the two tuples are different for the same  $A_i$  value, there is still at least one attribute that can discriminate the two tuples apart. The algorithm makes use of consistency checking to decide when to terminate the merging. Thus, the resulting data will not have any class conflict and will not reduce the discriminating power of the original data.

The major issues discussed in the paper include:

- **Starting point of merging** In our approach, the attributes to be merged are discretized in a round robin manner. We have an initial set consisting of all the attributes to be merged. Each attribute in the set is selected to have one interval merged before the next round of merging begins. When an attribute has finished merging it is deleted from the set. For a particular attribute, the starting point can be chosen either randomly, or in a particular order determined by attribute values, frequencies, and etc..
- **Order of target attributes** Due to the unique nature of the merging criterion, both the way of selecting the attributes in each round of merging and the order of choosing values to start an interval will influence the resulting intervals produced. Intuitively, attribute merged first will tend to have less intervals. However, the optimal ordering will be data dependent.

- **Noise in data** The algorithm implicitly eliminates the effect of noise present in the data. If a certain tuple contains noise in one attribute, then this noise is smoothed out by the merging criterion which also looks at the other attributes values. Also when the number of conflicted tuples are within the specified tolerance, the merging process continues.
- **Missing values** We decided that the new data values that falls between any two intervals are retained as it is without merging into either interval. Since we do not have any knowledge about the data value, the best thing to do is to retain the information to the greatest extent.

To test the effectiveness of the proposed algorithm, experiments were conducted using five data sets from the Machine Learning Repository of UCI. They are the IRIS (4 continuous attributes), GLASS ( 9 continuous attributes), WISCONSIN-BREAST-CANCER (20 continuous attributes), HEART DISEASE ( 13 non-continuous and 6 continuous attributes), and IONOSPHERE (33 continuous attributes). 10-fold cross validations were used in the experiments. C4.5 is used to test the effectiveness of the discretization with respect to the decision tree size and the classification accuracy. Since the target attribute and the starting point were both selected randomly, the discretizations were executed 200 times for each combination. The result obtained from the test is shown in Table 1.

**Table 1.** The effectiveness of CFMD

| Data   | Without disc. |              | With disc. |              | Improvement (%) |          |
|--------|---------------|--------------|------------|--------------|-----------------|----------|
|        | Tree size     | Accuracy (%) | Tree size  | Accuracy (%) | Tree Size       | Accuracy |
| Iris   | 8             | 93.3         | 8          | 94.1         | 0.00            | 0.86     |
| Glass  | 46            | 46.0         | 40         | 52.4         | 13.04           | 13.91    |
| Cancer | 20            | 94.8         | 16         | 96.0         | 20.00           | 1.27     |
| Heart  | 42            | 73.0         | 41         | 78.2         | 2.38            | 7.12     |
| Iono.  | 28            | 86.3         | 20         | 86.3         | 28.57           | 0.00     |

Table 1 gives the improvements achieved by discretization, which are calculated as the difference of the two measurements divided by the measurement for the case without discretization. It is interesting to note that, for certain data, e.g., the Glass data, C4.5 performs rather poorly when using the non-discretized data as input and the improvement provided by our method of discretization is the most significant (about 14%). For those data C4.5 provides high classification accuracy, discretization helps to reduce the tree size significantly (more than 20 % for Cancer and Iono data) while retaining the classification accuracy.

The effects of the ordering of attributes and the choosing of starting values were also tested. The experiments revealed that the ordering of attributes and the choosing of starting values does affect the resultant accuracy of classification. The average difference of the classification accuracy between different ordering of attributes and starting values both range from 13 % to 19%.

# Characteristic Rule Induction Algorithm for Data Mining

Akira MAEDA\*, Hideyuki MAKI\*, and Hiroyuki AKIMORI\*\*

E-mails: maeda@sdl.hitachi.co.jp, maki@sdl.hitachi.co.jp

\* Systems Development Laboratory, Hitachi, Ltd.  
1099 Ohzenji, Asao-ku, Kawasaki, 215 JAPAN

\*\* Device Development Center, Hitachi, Ltd.  
2326 Imai, Ohme-city, Tokyo, 198 JAPAN

## 1. Rule Induction Approach to Data Mining

The objective of data mining is to extract knowledge from large databases, mainly by applying inductive learning methodologies to real world data. Since the amount of data stored in information systems is rapidly increasing, there is a strong need for intelligent and powerful analysis techniques that can handle the huge data volume.

Our approach to data mining is to develop an algorithm that can extract useful regularities from large collection of data with incomplete, noisy, redundant variables, and possibly missing data values. Typical applications are database marketing, fault analysis, and diagnosis support systems.

In this paper, we describe a new characteristic rule induction algorithm named CHRIS (Characteristic Rule Induction by Subspace search), which is carefully designed for data mining. The algorithm is applied to a semiconductor fault analysis system for daily use, and produces effective results.

## 2. Characteristic Rule Induction Algorithm

### 2.1 Categorization

The basic idea of CHRIS is to extract rules in the form of “if  $A$  then  $B$ ” for the given target class  $B$ , by searching the conditional part  $A$  that describes the common features of instances belonging to  $B$ . The input data is considered to be a set of records. Each record consists of a set of fields. Field values can be either numerical or symbolic, and there may be missing values. The conditional part  $A$  is a combination of field values, such as “ $x_1$  is  $a_1$  and  $x_2$  is  $a_2$ ”.

First, numerical values are transformed to fuzzy categorical values, such as “*LARGE*” or “*SMALL*”. By this categorization, CHRIS can handle numerical and symbolic values in the same manner. The search for rules is performed by a standard “generate-and-test” strategy. The algorithm generates a candidate rule by combining from one through  $N$  categorical values to form a conditional part, where  $N$  is the number of fields. Then the algorithm evaluates the generated rule using the measure described in the next subsection.

### 2.2 Information Theoretic Rule Measure

The proposed algorithm uses “*utility measure*” to define the interestingness of rules. The algorithm extracts  $M$  rules that have largest utility measures among all possible rules.  $M$  is a number of rules to be extracted, and should be defined by the user. We do not take an approach to extract all the rules that have larger measures than a threshold, because a user cannot predict how many rules will be extracted, and also because extracted rules shown in the order of interestingness help a user quickly understand the regularities.

There are two measures to evaluate the usefulness of a rule “if  $A$  then  $B$ ”<sup>[2]</sup>. We call them *hit ratio* and *cover ratio*. The hit ratio measures how a rule is accurate. It can be defined as  $P(B|A)/P(B)$ , where  $P(B)$  is the probability that the conclusion part  $B$  is satisfied, and  $P(B|A)$  is the probability of  $B$  on the assumption that the conditional part  $A$  is satisfied. The cover ratio is defined as  $P(A)$ , which measures how a rule is generally applicable. Hit ratio and cover ratio are generally conflicting measures. Therefore, we define the utility measure by combining these two as:  $P(A) \cdot \log\{P(B|A)/P(B)\}$ . We introduce a free parameter  $\alpha$ , so that users can control the relative importance between hit ratio and cover ratio, according to the noise level of the input data and the user intention of the analysis.

### 2.3 Acceleration of Rule Search

We assume that the typical data size used for data mining is 1,000 to 100,000 records with more than 100 fields. The target performance is to deal with the data of this size in several minutes using average workstations and PC's. To achieve this, we have incorporated efficient pruning strategies in the algorithm.

CHRIS uses two criteria for the pruning, namely by the cover ratio and by the utility measure described in the previous subsection. Pruning by the cover ratio is straightforward. Since the cover ratio decreases monotonically with the levels within the tree, the search for subnodes is unnecessary when the cover ratio of the current node becomes less than the threshold.

Next, pruning by the utility measure is described. The goal of our rule induction algorithm is to extract rules with  $M$ -largest utility measures. After the algorithm calculate the utility measure of a rule (a node in the tree), the algorithm tries to find the upper limit of the utility measure of the nodes in the subtree of the current node. When the calculated upper limit is less than the threshold, which is the  $M$ -th largest value of the already found rules, the search in the subtree can be safely omitted.

## 3. Conclusions

The characteristic rule induction algorithm described in the previous section is implemented in a data mining tool *DATAFRONT* as a commercial software product. We have also applied the algorithm to several systems, such as semiconductor fault analysis, steel mill plant fault diagnosis, and database marketing systems. This algorithm has advantages in extracting nonlinear, robust, and easily understandable knowledge from real world data. The algorithm can be also applied to time-series data by combining with the pattern recognition techniques<sup>[5]</sup>. Extension to other types of data, such as text and image, remains a subject of further research.

## References

- [1] C. J. Matheus, et.al.: Systems for Knowledge Discovery in Databases, IEEE Transactions on Knowledge and Data Engineering, Vol.5, No.6 , pp.903-913(1993)
- [2] G. Piatetsky-Shapiro: Discovery, Analysis, and Presentation of Strong Rules, Knowledge Discovery in Databases, pp.229-248, AAAI Press(1991)
- [3] Y. Satoh, et.al.: A Data Mining Technique for Time Series Data, to be presented at the Ninth International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems (IEA/AIE'96) (1996)

# Data-Mining Massive Time Series Astronomical Data Sets – A Case Study

Michael K. Ng<sup>1</sup> Zhexue Huang<sup>2</sup> Markus Hegland<sup>3</sup>

<sup>1</sup> Department of Mathematics, The University of Hong Kong, Pokfulam Road, H.K.

<sup>2</sup> CRC for Advanced Computational Systems, Computer Sciences Laboratory, The Australian National University, Canberra, ACT 0200, Australia

<sup>3</sup> CRC for Advanced Computational Systems, CSIRO Mathematical and Information Sciences, GPO Box 664, Canberra, ACT 2601, Australia

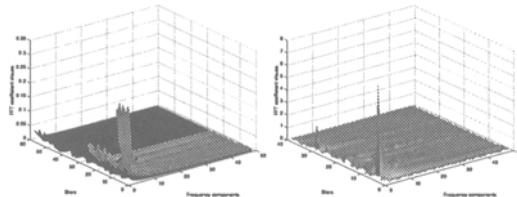
In this paper we present a new application of data mining techniques to massive astronomical data sets. Previous data mining applications deal with time-independent multiple spectral astronomical data [2]. We are concerned with time series astronomical data. More precisely, our data consists of  $N$  time series, each with a duration of  $M$  days. The data set can be viewed as an  $N \times M$  matrix in which rows are different time series and the ordered columns correspond to consecutive time points when measurements were made. Our primary objective of mining such data is to classify the time series according to their morphology and to identify classes which may carry some special signature of morphology.

The real data in this application consists of 40 million time series, each representing a sequence of brightness (light curve) of a star measured in one of two spectral bands on a daily basis in the MACHO project [1]. Totally, about 20 million stars (i.e.,  $N \approx 2 \times 10^7$ ) have been measured in the past four years. More than half a terabyte of time series data has resulted. The scientific discovery tasks in our data mining exercise are to discover new variable stars and to identify microlensing events represented in some star light curves.

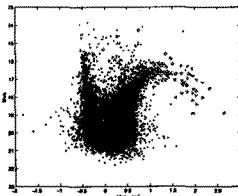
There are a number of big challenges in mining such a large time series database. The size of the data is an obvious one which is always of concern. The time dimension is another challenge which is not concerned in the other astronomical data mining applications [2]. The high dimensionality (i.e.,  $M \approx 1500$ ) and the varying phase factor of the time series make it hard to mine the data in the time domain. The potentially costly transformation of a large number of time series to some feature domain is inevitable. Selection of suitable features for appropriate representation of these time series in terms of conciseness and accuracy is one of the research problems. The data is far from error-free. Noise, invalid measurements and missing values exist in every time series. The burden of preprocessing and massaging the data is tremendous. All these problems have to be solved with different techniques before classification is performed.

The classification is based on dissimilarity of the star light curves. The distance measure in the frequency domain between two stars is used. After star light curves are converted to vectors in the feature spaces, we run the  $k$ -means algorithm hierarchically over a set of star feature vectors. We use some visual techniques to verify our clusters. To verify the similarity of stars within clusters and the dissimilarity of stars among clusters in the feature space, we display

the FFT coefficients of all stars in different clusters. The graph on the left side of Figure 1 shows three clusters produced based on the FFT coefficients. Each cluster is represented as a smooth surface which indicates that stars are similar in the clusters. One can also see that the three surfaces are quite distinguishable, which means that the stars in different clusters are quite dissimilar. The graph on the right side of Figure 1 shows a group of stars which were randomly selected from the variable stars. Because the stars in this group are not clustered according to dissimilarity measure, the surface of this group is very rough, which implies that the stars in this group belong to different clusters. We have also implemented other visualization tools to show star light curves in the time domain and found these tools are very useful in verifying clustering results and identifying variable stars. Figure 2 shows some variable stars identified using our clustering method and visualization tools from 360 variable candidates selected from about 20000 stars according to the variable defined by astronomers.



**Fig. 1.** The graph on the left side shows the plots of the FFT coefficients of stars in 3 clusters. The graph on the right side shows the plots of the FFT coefficients of stars in a randomly selected group.



**Fig. 2.** The color magnitude diagram of 20383 stars from the MACHO database. Each dot represents a star. The crosses are the candidates of variable stars determined by the variable index. The circles are the identified variable stars.

**Acknowledgments:** The authors wish to acknowledge that this work was carried out within the Cooperative Research Centre for Advanced Computational Systems established under the Australian Government's Cooperative Research Centres Program. The authors thank Dr T. Axelrod at MSSSO for providing us the sample data.

## References

1. T. S. Axelrod et al, "Statistical Issues in the Macho Project." Mt Stromlo and Siding Spring Observatories, The Australian National University, 1996.
2. U. M. Fayyad, S. G. Djorgovski and N. Weir, "Automating the Analysis and Cataloguing of Sky Surveys." in *Advances in Knowledge Discovery and Data Mining*, (eds) U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth and R. Uthurusamy, AAAI Press / The MIT Press, 1996, pp. 471–493.

# Multiple Databases, Partial Reasoning, and Knowledge Discovery

Chris Nowak

Macquarie University, Sydney NSW 2109, Australia

**Abstract.** We consider multiple sources of information, sets of sentences they provide, and the resulting multiple partial theories. The set of all theories is equipped with an information ordering and forms a lattice. We relate the framework to knowledge discovery.

## 1 Introduction

We consider a set of *sources of information*, and a set (*multiple database*) of sets of *sentences* the sources provide. A set of sentences (a single database) gives rise to its *logical closure*, or *theory*—given multiple information sources, a set of theories results. We investigate structures associated with multiple theories, and relate them to *knowledge discovery*.

Section 2 provides a concise presentation of a *partial reasoning* framework, while Section 3 relates the framework to *knowledge discovery*.

## 2 Partial Reasoning

Let  $M$  be a set of *attributes*,  $M \subseteq N$ , where  $N$  are natural numbers. Let  $G = \{g : M \rightarrow \{0, 1\}\}$  be a set of *partial objects*. Some subsets of  $G$  are set aside, and form a set  $\mathbf{W}$  of *partial worlds*. If  $W$  is a world then the (partial) objects of  $W$  indicate which sets of attributes are *exemplified* and which must not be.

Worlds can be *described* using a *language* introduced as follows. Let  $g \in G$ , and let  $\psi_g = \{m \mid g(m) = 1\} \cup \{-m \mid g(m) = 0\}$ . Then  $\Psi = \{\psi_g \mid g \in G\}$  is a set of *terms*, and  $\Phi = \{\oplus\psi, \ominus\psi \mid \psi \in \Psi\}$  is a set of *formulae*. We say that  $\oplus\psi$  is *valid* in  $W$  iff  $\exists_{g \in W} \forall_{m \in \psi} m \in \psi_g$ , and that  $\ominus\psi$  is *valid* in  $W$  iff  $\forall_{g \in W} \exists_{m \in \psi} -m \in \psi$ . Then, given a world  $W$ , let  $T_W = \{\varphi \mid W \models \varphi\}$  be called a *theory of the world*  $W$ ; certainly,  $T_W \subseteq \Phi$ . Let  $D \subseteq \Phi$ .  $D$  is a *description set* if there is a world  $W$  such that  $W \models D$  (i.e.,  $D \subseteq T_W$ ). A description set  $D$  gives rise to a formal system  $\mathcal{H}_D$ , with  $D$  being the set of *axioms*. The rules of inference are the following: (1) if  $\oplus\psi_2$  and  $\psi_2 \supseteq \psi_1$  then  $\oplus\psi_1$ ; (2) if  $\ominus\psi_1$  and  $\psi_1 \subseteq \psi_2$  then  $\ominus\psi_2$ ; (3) if  $\ominus\psi \cup \{m\}$  and  $\ominus\psi \cup \{-m\}$  then  $\ominus\psi$ ; (4) if  $\oplus\psi$  and  $\ominus\psi \cup \{m\}$  then  $\oplus\psi \cup \{-m\}$ . Given  $D$ , the corresponding *theory* is  $T = \text{Cn}(D)$ . Let  $\mathbf{T}$  be a set of all theories.  $\mathbf{T}$  is equipped with an *information ordering* relation  $\leq$ , defined by  $T_1 \leq T_2$  iff  $T_1 \subseteq T_2$ . Define  $T_1 \wedge T_2 = T_1 \cap T_2$ , and  $T_1 \vee T_2 = \text{Cn}(T_1 \cup T_2)$ , if consistent, or 1, otherwise.  $(\mathbf{T} \cup \{1\}, \leq)$ , or  $(\mathbf{T} \cup \{1\}, \wedge, \vee)$ , is a lattice (1 is stipulated to be the top element).

Given that some (partial) description sets are provided, one obtains a set  $B \subseteq T$  of theories.  $B$  gives rise to  $C = \text{Cl}_{\wedge, \vee}(B)$ , the closure of  $B$  under  $\wedge$ ,  $\vee$ . Then  $(C, \leq)$  is a *lattice*, and it is a sub-lattice of  $(T, \leq)$ .

The framework of *partial reasoning* is presented in [1].

### 3 Knowledge Discovery

Let  $\Phi$  be the formulae (sentences), and  $S$  be a set of information sources—then the information provided is  $\Delta \subseteq S \times \Phi$ . What *knowledge* can be *discovered*?

Perform: (1) Start with  $\Delta \subseteq S \times \Phi$ . (2) Find  $\{D_s\}_{s \in S}$ , where  $D_s \subseteq \Phi$ . (3) Find  $B = \{B \mid B = \text{Cn}(D_s)\}$ . (4) Find a set  $\Sigma$  of sets of equivalent (same theory) sources. (5) Treat  $B$  as  $\{B_\sigma\}_{\sigma \in \Sigma}$ . (6) Find  $C = \text{Cl}_{\wedge, \vee}(B)$ . (7) Note that  $(C, \leq)$  is a lattice, extend it to  $(C \cup \{1, 0\}, \leq)$ . (8) With every  $C \in C$  associate  $v_C = (v_C^+, v_C^-)$ , where  $v_C^+ = \{\sigma \in \Sigma \mid B_\sigma \geq C\}$  and  $v_C^- = \{\sigma \in \Sigma \mid B_\sigma \vee C = 1\}$ . (9) Treat  $(C, \leq)$  and  $\{v_C\}_C$  as *discovered knowledge*.

The proposed method of knowledge discovery is *order-theoretic*, appropriate for discovering knowledge from information provided by multiple sources. Indeed, according to [4, 5] “*knowledge discovery* is the nontrivial extraction of implicit, previously unknown, and potentially useful information from the data.”

### 4 Conclusion

The paper presents the framework of *partial reasoning* as a *knowledge discovery* process. Some of the related issues are (non-statistical) knowledge discovery with *rough sets* [2], *attribute exploration* [3], and discovering *association rules* [6].

### References

- Chris Nowak. Towards partial reasoning. In *The 11th International FLAIRS Conference*. Florida AI Research Society, AAAI Press, May 1998. To appear.
- Wojciech Ziarko. The discovery, analysis, and representation of data dependencies in databases. In Gregory Piatetsky-Shapiro and William J. Frawley, editors, *Knowledge Discovery in Databases*. AAAI Press / The MIT Press, 1991.
- Bernhard Ganter. Attribute exploration with background knowledge. In *Proceedings, ORDAL96*, 1996.
- Gregory Piatetsky-Shapiro. Knowledge discovery and acquisition from imperfect information. In Amihai Motro and Philippe Smets, editors, *Uncertainty Management in Information Systems. From Needs to Solutions*. Kluwer, 1996.
- William J. Frawley, Gregory Piatetsky-Shapiro, and Christopher J. Matheus. Knowledge discovery in databases: An overview. In Gregory Piatetsky-Shapiro and William J. Frawley, editors, *Knowledge Discovery in Databases*. AAAI Press, The MIT Press, 1991.
- Marcel Holsheimer, Martin Kersten, Heikki Mannila, and Hannu Toivonen. A perspective on databases and data mining. In *First International Conference on Knowledge Discovery and Data Mining, Montreal*, 1995.

# Design Recovery with Data Mining Techniques

Carlos Montes de Oca and Doris L. Carver

Department of Computer Science, Louisiana State University

298 Coates Hall, Baton Rouge, Louisiana, 70803. USA

(moca, carver)@bit.csc.lsu.edu

## 1 Introduction

Reengineering, reusing, and maintaining large legacy software systems require the use of design recovery techniques. Basically, the aim of design recovery is to produce meaningful high level abstractions that facilitate the understanding of the system [1]. These abstractions might take different forms such as formal specifications, module breakdown, and dataflows. In this paper, we discuss preliminary research and potential benefits of applying data mining (DM) techniques to the design recovery problem. This approach to design recovery derives from the observation that data mining can discover unsuspected non-trivial relationships among elements in large databases. This feature of data mining could be used to elicit new knowledge about a subject system. For example, data mining may uncover novel ways to form relationships among the system's components. Such information may be used to construct the design of the software system. In particular, our research explores the use of DM techniques to recover the design of imperative legacy code.

## 2 Applying Data Mining to Design Recovery

Our approach uses three steps: (1) generate a database representation of the legacy code, (2) mine this database, and (3) construct a design based on the outcome of the mining process. The first step involves defining a database view of the legacy system that contains enough information to allow data mining algorithms to extract relevant knowledge. The second step consists of selecting and applying data mining algorithms to the database view created in the previous step. The last step consists of combining the mined results into meaningful information to build a design.

We have started to explore these ideas. Our initial effort focuses on defining a database view of a subject system and on applying a data mining algorithm to this view. Particularly, our test model consists of a data base view that relates programs and files, and a DM algorithm that mines association rules [2]. Specifically, let  $S$  be a software system consisting of non-empty sets of programs  $P$  and files  $F$ . Let  $R$  be a relation and let  $a R b$  denote that  $a$  uses  $b$ . We build a database view of the system as a set of tuples. For each  $p \in P$  there is a tuple  $v = \{f \in F : p R f\}$ . That is, tuple  $v_i$  contains all the files that program  $p_i$  uses. Hence, the database view is the set  $V = v_1, v_2, \dots, v_{|P|}$ .

## 3 Case Study

We applied these ideas to a Point of Sale system. It consisted of 93 COBOL programs and 60 files. First, we identified and prepared the data required to build the data base view. We eliminated from the analysis the programs that use one file and the files that are used just once. Then, we assigned a unique number to each file and to each program. Second, we built the data base view accordingly to the

definition given above. The database view was represented as an ASCII file, which contained a tuple in each row. A typical tuple was  $<25\ 41\ 43\ 50\ 52>$ . That is, program 25 uses files 41, 43, 50, and 52. Finally, we used the Apriori [3] algorithm to mine association rules.

## 4 Results

We obtained several interesting associations. For example, the association  $<25\ [45\ 52]>$  means that 25 programs use files 45 and 52. If we consider that 50 programs use file 52 and 26 programs use file 45, then 96% of the programs that use file 45 also use file 52. That is, if file 45 is present in a program it implies that file 52 is also present in that program with 96% confidence. In short,  $[File\ 45] \Rightarrow [File\ 52]$  (0.96). Another association is  $<23\ [18\ 28]>$ . In this case, 28 programs use files 28, and 18. Therefore,  $[File\ 18] \Rightarrow [File\ 28]$  (0.82), and  $[File\ 28] \Rightarrow [File\ 18]$  (0.82). That is, this rule is commutative. Moreover, the association  $<18\ [45\ 48\ 52]>$  implies that  $[File\ 48] \Rightarrow \{[File\ 45] \wedge [File\ 52]\}$  (0.90) (note that 20 programs use file 48).

In general, our experiment produced novel knowledge about the system in the form of rules such as "file  $x$  is used along with file  $y$  in  $c\%$  of the programs that use file  $x$ ". Such an information might be used to identify clusters of files (i.e., modules). For example, the associations  $<13\ [45\ 48\ 50\ 52\ 53]>$  and  $<5\ [4\ 5\ 22\ 28\ 38\ 45\ 48\ 50\ 52\ 53\ 58]>$  suggest that files 45, 48, 50, 52, and 53 might belong to the same module because they stay together in 13 programs. Moreover, 5 programs use these files in combination with files 4, 5, 22, 28, 38, and 58, suggesting that this last set of files belongs to other module and that those 5 programs access files from 2 modules.

## 5 Conclusions

We presented an approach to software design recovery using data mining tools. This approach offers the advantage of scalability. In fact, the larger the software being analyzed the better. Moreover, this approach is capable of discovering associations among system's components regardless the naming conventions and programming style. Finally, it is capable of producing new ways to relate system's components.

This initial experience shows that applying data mining techniques to design recovery conveys the following three challenges: the selection of a database representation of the subject system, the selection of the data mining technique, and the consolidation of the mined knowledge into a meaningful design. Our preliminary results are promising. We have been able to detect novel relationships among the system's files. The current challenge is to combine these results to create the system's design.

## References

1. Chikofsky, E.J., Cross II, J.H., "Reverse Engineering and Design Recovery: A Taxonomy," *IEEE Software*, Vol. 7, No 1 (January 1990), pp. 13-17.
2. R. Agrawal, T. Imielinski, A. Swami, "Mining Association Rules between Sets of Items in Large Databases," *Proc. ACM Int'l Conf. Management of Data*, pp. 207-216, May 1993.
3. R. Agrawal, R.Srikant, "Fast Algorithms for Mining Association Rules in Large Databases," *Proc. 20<sup>th</sup> Int'l Conf. on Very Large Data Bases (VLDB '94)*, pp.478-499, Santiago, Chile, Sep. 1994.

# The CLARET Algorithm

Adrian R. Pearce<sup>1</sup> and Terry Caelli<sup>2</sup>

<sup>1</sup> School of Computing, Curtin University, Perth WA, Australia

<sup>2</sup> Center for Mapping, The Ohio State University, Columbus, Ohio, USA

The Consolidated Algorithm for Relational Evidence Theory (CLARET) is used for interpreting actions and events in dynamical domains. This includes a data mining application for recognising and describing aeroplane manoeuvres and pilot intentionality in agent-oriented flight mission modelling. Another is an on-line schematic interpretation system for hand-drawn symbols and equations. The aim of this work is to bind descriptions to spatio-temporal "trajectories" present in time-series output. For example in describing flight, an approach to land manoeuvre is defined by the subsequence of different roll-pitch-yaw states of the aeroplane and different actions on the control yoke. This extends work on behavioural cloning in flight simulators [8] and is currently being used for binding trajectory information to symbolic belief, desires and intentionality planning systems [3]. In hand-drawn schematic interpretation, diagrams are defined by different strokes or gestures, drawn at different times at particular orientations. We have developed a software package—the Consolidated Learning Algorithm (CLARET) [4]—which is based on Relational Evidence Theory [5]. The package adapts relational learning techniques [6, 2] to utilise the constraints present in time series data—those of states and their continuous valued, attributed relationships.

In the CLARET algorithm, an unknown segmented and labelled trajectory is presented to the system together with  $n$  known trajectories. First, relational descriptions are generated by extracting relationships between pattern segments. Pen trajectories are first segmented into line-based descriptions using a hierarchical multi-scale polygonal approximation method. Features are then extracted from individual states in continuous numerical form. Second, the patterns are matched using the attribute values and labelled parts from relational descriptions. It involves the repetition of the following steps: First, Attribute generalisation by partitioning numeric attributes of relations. Second, Graph matching: matching parts defined by attribute bounds generated in first step. Third, Relational specialisation: adding relations to resolve uncertainty. The purpose of relational specialisation is to generate conditional rules by adding literals to the current clause. This creates paths through the relational structures. The method is based on Conditional Rule Generation [1] and labelled, first order rules are generated in the form,

$$r_i(P_J, P_K, A_1, A_2, \dots) \leftarrow r_j(P_I, P_J, A_1, A_2, \dots), 0.25 \geq A_1 \leq 0.95, \dots$$

where  $P_I$  are part variables and  $A_i$  are attributes. Note that label compatibility is represented via paths through relational structures of arbitrary arity  $P_I, P_J \rightarrow P_J, P_K \rightarrow \dots$ , which instantiate conditional rules  $r_i, r_j, \dots$  hierarchically.

The method used for generalising over attributes is based on the attribute selection and splitting technique used in decision trees, notably C4.5 [7], except that the information metric is replaced by a variance criterion. Attributes are partitioned into regions and each split results in two new regions defined by the conjunctions of attribute bounds based on minimising the variance criterion. Each partitioned rule is replaced by two new least general rules, rule  $r_0$  is replaced with rules  $r_1$  and  $r_2$ ). Such hashing of the observed relational attributes provides initial hypothesis about the resolution of parts required to recognise and discriminate between the known patterns.

The question can now be asked: are these rules specific enough to differentiate between the different relational structures present to correctly interpret the current pattern? Graph matching is used to solve this problem of mapping current (unknown) line segments to existing (known) line segments, by finding compatible sets of labelled rules, using Rulegraphs [5]. In addition, CLARET uses a relational evidence network to deterministically order the search resulting in an admissible strategy. Relational Evidence Theory extends the Bayesian conditioning framework to incorporate checks for label compatibility between rules. The search is made tractable by reducing the cardinality of the search space using multi-labelled rulegraph interpretations together with dynamic programming principles and relational evidence measures to prune the search.

The significance of this work are twofold. First, it extends the applicability of machine learning theories and algorithms for data mining in continuous valued temporal domains. Second, it complements the query tools currently available in processing time-series data, by exploring new ways of representing and generalising over temporal information.

## References

1. Bischof, W. F., Caelli, T.: Learning Structural Descriptions of Patterns: A New Technique for Conditional Clustering and Rule Generation. *Pattern Recognition* **27** (1994) 689–97
2. Bratko, I., Muggleton, S.: Applications of inductive logic programming. *Communications of the ACM* **38** (1995) 65–70
3. Georgeff, M. P., Lannsky, A.: Procedural Knowledge. *Proceedings of the IEEE (Special Issue on Knowledge Representation)* **74** (1986) 1383–1398
4. Pearce, A., Caelli, T., Bischof, W.: CLARET: A new Relational Learning Algorithm for Interpretation in Spatial Domains. *Proceedings of the Fourth International Conference on Control, Automation, Robotics and Vision (ICARV'96)*, Singapore, December (1996) 650–654
5. Pearce, A. R., Caelli, T., Bischof, W. F.: Rulegraphs for Graph Matching in Pattern Recognition. *Pattern Recognition* **27** (1994) 1231–47
6. Quinlan, J. R.: Learning logical definitions from relations *Machine Learning* **5** (1990) 239–66
7. Quinlan, J. R.: C4.5 Programs for Machine Learning Morgan Kaufmann (1993)
8. Sammut, C., Hurst, S., Kedzier, D., Michie, D.: Learning to Fly *Proceedings of the Ninth International Conference on Machine Learning*, Morgan Kaufmann (1992) 385–393

# LRTree: A Hybrid Technique for Classifying Myocardial Infarction Data Containing Unknown Attribute Values

Christine L. Tsien, S.M.<sup>1,2</sup>, Hamish S. F. Fraser, M.R.C.P., M.Sc.<sup>2,3</sup>,  
Isaac S. Kohane, M.D., Ph.D.<sup>1,4</sup>

<sup>1</sup> Harvard Medical School, Boston, MA

<sup>2</sup> Laboratory for Computer Science, M.I.T., Cambridge, MA

<sup>3</sup> Div. of Clinical Decision Making, Tufts-New England Medical Center, Boston, MA

<sup>4</sup> Children's Hospital Informatics Program, Children's Hospital, Boston, MA

## 1 Introduction

A common obstacle in using knowledge discovery techniques is the occurrence of unknown attribute values in the data[3]. This can make difficult not only the application of traditional models, but also the comparison of competing models that include slightly different sets of attributes. A technique combining aspects of decision trees and logistic regression (LR), named "LRTree," has been developed for handling missing data when very few values per case are unknown. Where no values are absent, the model behaves like a traditional decision tree. For cases in which one or more attribute values are unknown, as much of the underlying decision tree as possible is followed. Once a branch point arises for which the value is unknown, an LR equation associated with that branch, specifically built without the unknown attribute(s), is employed. The idea for this hybrid technique is reminiscent of White's "dynamic path generation" model[5] in which a path of a decision tree is built for each case with unknown attribute values (compared to use of an LR equation here). The potential advantage of the LRTree lies in the deterministic nature of deriving an LR equation. In contrast, from that same set of data, numerous decision trees can be induced; development of an optimal path at each step should thus require human input. Moreover, an LR equation derived from a tree's training data and including exactly those attributes present in the tree was shown to yield statistically equivalent classification results with that tree[4].

The associated LR equations can be derived from either the full training set used in tree induction, or from the partial training sets that originally reached each node in the tree. Both of these possibilities have been explored and evaluated on two large clinical data sets in the domain of diagnosing myocardial infarction in patients who present to an emergency room with chest pain[2, 4].

## 2 Methods

The decision tree backbone of the LRTree used for this experiment, FT Tree[4], was developed with C4.5 using a 480-case chest pain data set from Edinburgh,

Scotland[1]. Another 622 cases from the same hospital, as well as 500 cases from Sheffield, England, served as test sets. Training data was left entirely intact, while both systematic and random deletion of data items (up to one attribute value per test case) were performed on each test set.

The 480-case Edinburgh training set was first partitioned via flow through the FT Tree to create data subsets. For each subset, an LR equation of up to nine attributes each was derived. The purposely omitted tenth attribute corresponds to the test attribute of the node from which the data subset came. Additionally, ten LR equations, in which exactly one attribute was omitted from each, were derived from the full 480-case data set. This resulted in two LRTree models: FT480 LRTree, which consists of the underlying FT Tree associated at each branch point with the appropriate LR equation derived from the 480-case set; and FT LRTree, which is associated at each branch point with the appropriate LR equation derived from the partitioned training subsets. (“Appropriate” LR equation refers to that which omits the test attribute of the current node.)

The same experiments were performed using an “LRSet:” a set of eleven LR equations (ten LR equations containing nine attributes each and one ten-attribute equation, all derived from the 480-case set). During classification, if all relevant values are known, the ten-attribute equation is applied; when one attribute value is unknown, the appropriate nine-attribute equation is applied.

### 3 Results

Table 1 lists the areas under the receiver operating characteristic (ROC) curve (calculated by trapezoidal estimation) for each of the three models run on each of the two test sets.

**Table 1.** Areas under the ROC Curve for Three Models.

| Omitted attribute | EDINBURGH TESTS |              |        | SHEFFIELD TESTS |              |        |
|-------------------|-----------------|--------------|--------|-----------------|--------------|--------|
|                   | FT LRTree       | FT480 LRTree | LRSet  | FT LRTree       | FT480 LRTree | LRSet  |
| age               | 94.04%          | 94.26%       | 94.74% | 88.54%          | 89.39%       | 88.91% |
| right arm pain    | 93.96%          | 94.12%       | 95.07% | 89.28%          | 89.88%       | 89.56% |
| pain duration     | 93.10%          | 94.09%       | 94.63% | 89.02%          | 89.51%       | 88.94% |
| crackles          | 94.15%          | 94.14%       | 94.42% | 89.34%          | 89.88%       | 87.60% |
| ST elevation      | 85.55%          | 85.55%       | 85.55% | 76.95%          | 76.95%       | 76.95% |
| new Q waves       | 74.83%          | 93.94%       | 94.02% | 63.49%          | 89.58%       | 89.40% |
| family history    | 94.07%          | 94.67%       | 94.83% | 87.02%          | 89.30%       | 89.42% |
| ST depression     | 43.73%          | 75.41%       | 80.74% | 48.52%          | 83.14%       | 85.61% |
| T wave changes    | 93.67%          | 94.17%       | 94.32% | 88.36%          | 89.46%       | 88.92% |
| old ischemia      | 94.04%          | 94.55%       | 94.15% | 88.46%          | 89.46%       | 88.72% |
| random #1         | 91.65%          | 93.61%       | 94.40% | 87.20%          | 88.86%       | 88.09% |
| random #2         | 91.74%          | 94.24%       | 93.60% | 87.15%          | 89.58%       | 88.56% |
| random #3         | 90.69%          | 93.22%       | 94.11% | 86.77%          | 88.94%       | 88.41% |
| averages          | 87.32%          | 92.00%       | 92.66% | 82.32%          | 87.99%       | 87.62% |

## 4 Discussion

Although on the Edinburgh test sets, the LRSet method tended to perform best, the FT480 LRTree tended to perform best for both systematically and randomly deleted data on the Sheffield tests. Because performance on Sheffield tests is more predictive of model generalizability, this result is promising. The FT480 LRTree likely performs better than the FT LRTree because some of the latter's LR equations are derived from extremely small sets of data. With more training data, it might be found that the LRTree with LR equations derived from partitioned data subsets actually performs as well as or even better than the LRTree using LR equations derived from the entire training set.

For domains in which more than one unknown attribute is present, the LRTree technique can be modified such that each branch point of the underlying tree becomes associated with a *set* of LR equations, each one able to handle a different subset of attributes. Clearly, however, this could lead to generation of an exponentially large number of LR equations. One possibility would be to only generate LR equations on-the-fly as needed, optionally storing each equation to avoid duplication of effort.

Decision tree-based clinical aids are advantageous due to their intuitiveness and understandability, features that must be offered if clinicians are actually to adopt such aids into practice. The preliminary results have indicated that in this type of domain, in which few attribute values will be unknown, the LRTree technique may be a viable option for making more robust such decision aids.

## Acknowledgments

The authors would like to thank P. Szolovits, J. Doyle, W.J. Long, R.L. Kennedy, and the Howard Hughes Medical Institute. Portions of this work were supported by DARPA under contract F30602-97-1-0193.

## References

1. Kennedy RL, Burton AM, Fraser HS, *et al*: Early diagnosis of acute myocardial infarction using clinical and electrocardiographic data at presentation: derivation and evaluation of logistic regression models. *EHJ*. (1996) 17: 1181-1191
2. Long WJ, Griffith J, Selker H, D'Agostino R: A comparison of logistic regression to decision-tree induction in a medical domain. *Comput Biomed Res* (1993) 26: 74-97
3. Quinlan JR: Unknown attribute values in induction. *Proceedings of the Sixth International Workshop on Machine Learning*. Segre AM, editor. (1989) 164-168
4. Tsien CL, Fraser HS, Long WJ, Kennedy RL: Using classification tree and logistic regression methods to diagnose myocardial infarction. To appear in *MedInfo '98*.
5. White AP: Probabilistic induction by dynamic path generation in virtual trees. In: Bramer MA; ed. *Research and Development in Expert Systems III*. Cambridge: Cambridge University Press. (1987): 35-46

# Modelling Decision Tables from Data

G. WETS, J. VANTHIENEN

Katholieke Universiteit Leuven

Department of Applied Economic Sciences  
Naamsestraat 69, 3000 Leuven, Belgium

H. TIMMERMANS

Eindhoven University of Technology

Department of Architecture & Urban Planning  
P.O. Box 513, Mail station 20, NL-5600 MB  
Eindhoven, The Netherlands

## ABSTRACT

On most datasets induction algorithms can generate very accurate classifiers. Sometimes, however, these classifiers are very hard to understand for humans. Therefore, in this paper it is investigated how we can present the extracted knowledge to the user by means of decision tables. Decision tables are very easy to understand. Furthermore, decision tables provide interesting facilities to check the extracted knowledge on consistency and completeness. In this paper, it is demonstrated how a consistent and complete DT can be modelled starting from raw data. Because the modelled decision tables are sufficiently small they allow easy consultation of the represented knowledge.

## 1 Decision tables

A DT is a tabular representation used to describe and analyze procedural decision situations, where the state of a number of conditions jointly determines the execution of a set of actions. Not just any representation, however, but one in which all distinct situations are shown as columns in a table, such that every possible case is included in one and only one column (completeness and exclusivity). The tabular representation of the decision situation is characterized by the separation between conditions and actions, on one hand, and between subjects and conditional expressions (states), on the other. Every table column (decision column) indicates which actions should (or should not) be executed for a specific combination of condition states. In this definition, the DT concept is deliberately restricted to the single-hit table, where columns are mutually exclusive. Each possible combination of conditions can be found in one and only one column. Only this type of table allows easy checking for consistency and completeness (Vanthienen and Dries, 1997). If it is necessary, columns in an expanded DT can be contracted. Contraction is important in order to enhance the effectiveness of the decision-making or to provide a more compact formulation that can serve as a basis for discussion between the expert and the knowledge engineer. An example of a contracted DT is depicted in Fig. 1.

| 1. Costs (C) | C<2              |       | 2<=C<4 |                   | C>=4 |
|--------------|------------------|-------|--------|-------------------|------|
| 2. Space (S) | S<20 or 20<=S<40 | S>=40 | S<20   | 20<=S<40 or S>=40 | -    |
| 1. Premium 1 | -                | -     | -      | x                 | x    |
| 2. Premium 2 | x                | -     | x      | -                 | x    |
|              | 1                | 2     | 3      | 4                 | 5    |

Fig. 1. Example of a contracted DT

## 2 Modelling DTs from data

To model a DT three information elements are necessary: conditions, actions and the decision logic. First, conditions, actions and their respective states have to be retrieved from the dataset. After that the conditions and the actions are obtained the decision logic has to be derived. In a classical DT modelling method, the decision logic is elicited from an expert (e.g., in the form of rules) and subsequently these rules are used to model the DTs (Vanthienen & Wets, 1994). In this paper, however, the decision logic will be extracted from the dataset using some kind of classification technique and then it will be imported into the DT. Several hypothesis spaces can be used to model the decision logic (e.g. rules and decision trees). In our experiments, we used C4.5 (Quinlan, 1993) to obtain the decision logic. C4.5 is a well-known example of a classification tree algorithm. This type of algorithms tries to fit a tree to a training sample using recursive partitioning. This means that the training set is split into increasingly homogeneous subsets until the leaf nodes contain only cases from a single class. After that the decision tree is obtained, C4.5 allows to transform the decision tree in rules. These rules can be used to model the DT, as will be explained next.

At this point in the development process two major options can be chosen: constructing a DT from the decision logic which reflects all the information present in the decision logic (thus, also, several types of anomalies such as inconsistency and incompleteness); or constructing a DT which is consistent, complete and correct. When the former option is chosen, the expert himself has to decide how the anomalies which are presented in the DT have to be resolved. The latter option proposes a DT with no anomalies to the expert. To construct such a DT, anomalies which are present in the DT have to be removed using some heuristic. However it is not only necessary that those anomalies are not reflected anymore in the DT, but also the constructed DT should be as correct as possible. Of course it is clear that a DT which is completely correct cannot be constructed for real-life problems because the induced decision logic only partly represents that correct hypothesis. Both options can be easily combined. For each anomaly in the DT the system can make a suggestion, but it is up to the expert to decide whether he agrees to this suggestion.

## 3 Conclusion

Originally, DTs were constructed based on some knowledge provided by an expert or some piece of regulation. In this paper, we have demonstrated that it is possible to model a complete and consistent DT from data. Our proposed approach was empirically validated and it was shown that the modelled DTs are small enough in order to facilitate consultation.

## 4 References

- Quinlan, J. R. (1993), *C4.5 Programs for Machine Learning*, Morgan Kaufmann Publishers, San Mateo (CA).
- Vanthienen, J. & Dries, E. (1997), Decision tables: Refining concepts and a proposed standard, *Comm. of the ACM*, to appear.
- Vanthienen, J. & Wets, G. (1994), From decision tables to expert system shells, *Data & Knowledge Engineering 13*, pp. 265-282.

# A Classification and Relationship Extraction Scheme for Relational Databases Based on Fuzzy Logic

M. Vazirgiannis

Knowledge & Database Laboratory, Dept. of EI. & Computer Engineering,  
National Technical University of Athens, 15773 Athens – HELLAS  
(michalis@dbnet.ntua.gr)

## 1. INTRODUCTION

In traditional data classification and mining approaches [Sri96] the objective is the extraction of association among attribute values (e.g.  $A_i.X \rightarrow A_j.Y$ ) and assign some metrics that quantify the strength or the interest of the association. A widely accepted scheme for such metrics is the set of  $\{support, confidence\}$  measures where the former corresponds to the percentage of the total values of  $A_i$  that are equal to  $X$ , while the second reflects the percentage of this set that has  $Y$  as value for  $A_j$ .

This scheme does not cover properly the semantic properties of quantitative attributes [Sri96]. Assume the transaction log of a computer sales store and the subset of its schema:  $R = \{\text{client\_salary}, \text{client\_age}, \text{price}, \text{date\_of\_p}, \text{time\_of\_p}\}$ . Applying the techniques proposed in [Sri96] we would come up with rules of the form:

*client\_salary[3000,4500] and client\_age[25-40]  $\Rightarrow$  price[2500,3000]*

Apparently no manager would submit such a query, as non-computer experts are more interested in submitting close to natural language queries. On the other hand such a rule would not give the slightest idea to a manager/analyst of its importance, since it does not place the rule in the greater context of the involved attributes, i.e. what does *client\_salary[3000,4500]* mean in the full range of salaries and also in conjunction to its statistical distribution features. The result is that many “interesting” tuples are rejected due to the crisp limits that have been set. It is evident that the users would like to interact with a database in terms of rules and queries close to natural language, for instance:

*client\_salary high and client\_age young  $\Rightarrow$  price medium*

These natural language expressions should be mapped to the underlying database through a layer that maps the natural language terms to the underlying database schema and values. Our aim in this research effort is: i) the definition of a *classification* scheme of non categorical attributes into lexically defined categories based on fuzzy logic, ii) the definition of a *relationship extraction scheme* between attributes based on the above mentioned classification scheme.

## 2. THE PROPOSED SCHEME

The issue of *classification* involves the definition of categories that group the values of an attribute  $A$  in sets that have a specific feature. Let's assume the attribute “*client\_salary*” which in a data set ranges between 100 and 500. In real world people are talking about *high*, *low*, *moderate* salaries. How would one classify a specific salary value in to a category? What are the value-ranges corresponding to these categories? Are they overlapping? As it is clear, there is inherent uncertainty in the classification of a value in a set of categories. A fundamental issue is the acquisition of the related knowledge i.e.: the categories, the corresponding value ranges, the mapping functions between the real values and the fuzzy domain.

We aim at definition of a classification and relationship extraction framework to support natural language queries and assessments. In this effort we exploit fuzzy logic methodologies [Zad68] as tools for representation and manipulation of the classification uncertainty. The classification scheme is applied on a data set  $S$  under a certain relational

schema  $R = \{A_i\}$  where  $A_i$  is an attribute. The values of the non-categorical attributes ( $A_i$ ) are classified into categories according to a set of lexical values  $L = \{l_i\}$  (where  $l_i$  a lexical value corresponding to a classification category) each one for and a set of classification functions based on fuzzy logic methodologies. The result of this procedure is a set if degrees of belief (d.o.b.s)  $M = \{\mu_{l_i}(t_k, A_i)\}$  that a specific value  $t_k, A_i$  (where  $t_k$  is the tuple identifier) belongs to a the set denoted by the lexical value  $l_i$ . The d.o.b. that this value belongs to the sets denoted by the lexical values and the corresponding domains is:

$$\mu_{l_i}(t_k, A_i) = f(t_k, A_i)$$

where  $f$  is the transformation function. This function maps the value to the  $t_k, A_i$  fuzzy domain mentioned before. The choice of functions is a fundamental issue and will have a great impact on the credibility of the d.o.b.s. Thus a cube-like solid  $C$  is formed (see Figure 1). In each cell  $C[A_i, l_i, t_k]$  we store the corresponding d.o.b.

Then we need to have an accumulative measure of the information that is included in the values of the attribute as regards each lexical value. There we exploit the energy metric function [Gup88]. For instance, the overall belief that the a corresponding data set contains *low prices* is given by:

$$E_{\text{price} = "low"}(S) = \sum \left[ \mu_{\text{price} = "low"}(v) \right]^q$$

where  $v$  is a value of the attribute price and  $q$  is a non negative integer.

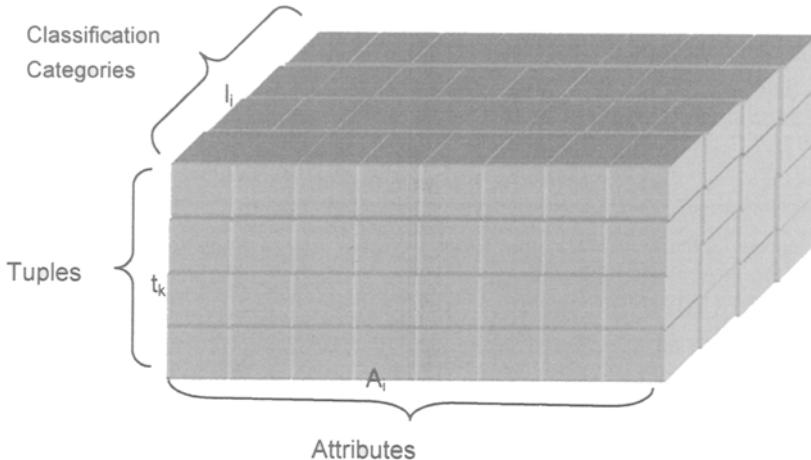


Figure 1. The “degree of belief” (d.o.b.) cube for attributes values and classification categories.

We may further compare the energy metric of different sets of data over such a criterion. For instance let  $a_i$  an attribute and  $l_i$  a lexical value, and two data sets  $S_1$  and  $S_2$ , then the corresponding energies are:  $E_{a_i=l_i}(S_1)$ ,  $E_{a_i=l_i}(S_2)$  indicating to which degree each data set fulfills the criterion  $a_i=l_i$ .

The significance of the classification scheme as regards the categories and is represented by the *energy of each lexical value*  $l_i$  of an attribute  $A_i$  given by the formula:

$$E_{l_i}(A_i) = \sum_k [\mu_{l_i}(t_k, A_i)]^q$$

This energy shows how significant is the information contained in the values of this attribute and also how well the data set fits to the classifications scheme and vice versa. The *overall energy* of an attribute  $A_i$ , would then be the sum of the energy metric values for all the attribute categories. This would express the overall information energy included in the values of the attribute (i.e. how strong is the belief for the classification assessment) and also the amount of information as regards the considered lexical values. Hence:

$$E_{Ai} = \sum_{li} [E_{li}(A_i)]$$

Based on the classification framework presented in the previous section we hereafter present a scheme for extraction of relationships among attribute values classifications. The relationships can be retrieved either as results of queries or as a result of data mining procedure for rule extraction. Here it is important to stress that we are searching for association between value classifications and not between values. Alternatively we might want to classify a combination of values from a tuple. In this case we need to somehow combine the corresponding d.o.b.s in a function. For instance

$$\mu_{\text{price}=\text{"low"} \wedge \text{time}=\text{"morning"}(t_k.A_i, t_k.A_j)} = f(\mu_{\text{price}=\text{"low"}(t_k.A_i)}, \mu_{\text{time}=\text{"morning"}(t_k.A_j)})$$

where  $\diamond$  is the relationship among the two classification predicates. It may range among the operators : “and”, “or” etc. Having such classification possibilities we may query our database over a variety of multiple criteria regarding the participating attributes. For instance we may submit the query: “*What is the overall belief that the database contains transactions for cheap purchases made in the morning ?*”

Abstracting we would state that the energy of the data set as regards the lexical values  $li$ ,  $lj$ , and the relationship  $l_i \diamond l_j$  would be:

$$E_{li \diamond lj}(A_i) = \sum_k [\mu_{li \diamond lj}(t_k.A_m), (t_k.A_z)]^{\frac{1}{2}}$$

which represents the belief that tuple  $t_k$  has both features (belongs to both categories)  $li$ ,  $lj$  and therefore it is classified accordingly.

### 3. CONCLUSIONS

We feel there is a lot of potential in the area of mining association rules with regards to classification of quantitative attributes. In this work we presented a scheme for: i) classification of database values into categories characterized by lexical values, ii)mining of association rules between these classifications being able to express high-level natural language queries and rules.

Further work will be concentrated in the following directions: i) manipulation of rules that are extracted from this scheme. Attached to that is the definition of the corresponding “confidence” and “support” attributes for the rules, ii) organizing the uncertainty values in appropriate Data Warehouses applying the appropriate indexing schemes.

### 4. REFERENCES

- [Gup88] Gupta, M.M., and Yamakawa, T., 1988, (editors), Fuzzy Logic and Knowledge Based Systems, Decision and Control (North-Holland).
- [Sri96] Srikant R., Agrawal R., “Mining Quantitative Association Rules in Large Relational Tables”, in the proceedings of ACM-SIGMOD ’96 Conference.
- [Zad68] Zadeh L. A., “Probability measures of fuzzy events”, Math. Anal. Applications, 23:421-427, 1968

# Mining Association Rules for Estimation and Prediction

Takashi Washio, Hiroki Matsuura\* and Hiroshi Motoda

Institute of Scientific and Industrial Research, Osaka University  
8-1 Mihogaoka, Ibaraki, Osaka, 567, Japan  
[washio@isir1.sanken.osaka-u.ac.jp](mailto:washio@isir1.sanken.osaka-u.ac.jp)

## 1 Introduction

The standard Basket Analysis derives all frequent itemsets and all association rules having support and confidence levels greater than their thresholds, and filters out trivial rules in statistical sense (1994, 1991). This framework gives comprehensive "*descriptions*" of regularities contained in the data. Another major purpose of data mining is to derive important knowledge for "*estimation and prediction*" on the underlying system which has generated the data (1996). We propose a novel principle to derive association rules for the latter purpose, where the rules provide maximal guesses from minimal facts about the system while maintaining their support and confidence levels as uniform as possible. The principle and its evaluation through real world data are described in the later sections.

## 2 Criteria for Estimation and Prediction

Given a set of transactions each of which consists of some items, the Basket Analysis derives "*association rules*" having the following form where "*Body*":  $B$  stands for an itemset and "*Head*":  $H$  another itemset (a superset of  $B$ ).<sup>2</sup>

$$B \Rightarrow H, \text{ where } B \subset H.$$

The "*support*" values of  $B$  and  $H$ , i.e.,  $sup(B)$  and  $sup(H)$ , are ratios of the number of transactions including each set to the total number of transactions respectively. The "*confidence*" value,  $conf(B \Rightarrow H)$ , stands for the credibility of the rule, and is defined as a ratio of the number of transactions including  $H$  to the number of transactions including  $B$ . The itemset having its support value greater than a threshold  $l - sup$  is called a "*frequent itemset*" (1997).

The standard Basket Analysis generates all association rules, where its head is a frequent itemset, and its confidence value is greater than another threshold value  $s - conf$ . Furthermore, some statistical "*rule-filters*" are adopted to the generated rules to extract only the rules "*describing*" interesting features embedded in the objective data (1991).

---

\* Current affiliation is Fujitsu Kansai Communication Systems Ltd.

<sup>2</sup> This representation of association rules is different from the standard notion  $B \Rightarrow R$  where  $R = H - B$ . We use  $H$  instead of  $R$  for ease of our explanation.

On the other hand, our interest is to establish a novel principle to generate and extract association rules for "*estimation and prediction*" on the system underlying the data. First, we propose the following criteria for the basis.

**Support threshold:** The head of every association rule must have the support greater than a threshold "*lowest support*":  $l - sup$ .

**Uniform confidence:** Every association rule must have a confidence close to but not less than a level "*specified confidence*":  $s - conf$ .

**Maximal estimation:** Every association rule must estimate a maximally specific consequence from a minimal fact.

The following definitions are introduced to implement these criteria.

**Minimal bodyset :** For a specified confidence  $s - conf$ , if a rule  $B \Rightarrow H$  satisfies the following condition,  $B$  is said to be a "*minimal bodyset*" of  $Head : H$  under  $s - conf$ .

$$conf(B \Rightarrow H) \geq s - conf \text{ and } conf(B' \Rightarrow H) < s - conf \quad \forall B' \subset B$$

**Maximal headset :** For a specified confidence  $s - conf$ , if a rule  $B \Rightarrow H$  satisfies the following condition,  $H$  is said to be a "*maximal headset*" of  $Body : B$  under  $s - conf$ .

$$conf(B \Rightarrow H) \geq s - conf \text{ and } conf(B \Rightarrow H') < s - conf \quad \forall H' \supset H$$

**Maximal estimation rule :** For a specified confidence  $s - conf$ , if  $Body : B$  and  $Head : H$  of a rule  $B \Rightarrow H$  are the minimal bodyset and the maximal headset respectively,  $B \Rightarrow H$  is said to be a "*maximal estimation rule*".

The maximal estimation rule satisfies the aforementioned criteria.

The maximal estimation rules contain some redundancy in terms of the estimation and prediction. We apply a logical "*rule-filter*" where the rule  $AB \Rightarrow ABR$  is removed, when two maximal estimation rules

$$AB \Rightarrow ABR \text{ and } B \Rightarrow BCR$$

are obtained. Here, every intersection among  $A, B, C, R$  is empty, and  $AB = A + B$ ,  $ABR = A + B + R$  and  $BCR = B + C + R$ . This rule-filtering does not violate the criteria of support threshold and uniform confidence.

### 3 Evaluation through call tracking data

The practical performance of our proposing framework has been evaluated in comparison with the standard approach (1994, 1991). The data used are a set of real call tracking data acquired in a telecommunication company in which each call of a person is recorded as a transaction (1996). Total number of the transactions involved is 65,525, where each item stands for calling date, calling type (oral, cellular, data, and busy), names of cities of calling and called persons, marital status and sex of those persons. Totally 221 types of items appear.

Table 1 shows the results of the evaluation. The numbers of generated rules before the rule-filtering indicated in the second column show the efficient reduction of the rules in our method because it enumerates the maximal estimation rules only. The numbers of filtered rules in the third column are also smaller in our method. The forth column shows the ratio of the average cardinalities of the bodyset and the headset, i.e.,  $[ave. card. of headsets]/[ave. card. of bodysets]$ , in each method. We have also checked the union of bodysets and the union of headsets of all rules for each method in every condition, and found the complete agreement between our method and the standard. These results indicate the higher ability of each rule derived in our method for estimation and prediction. The fifth column indicates the average and the standard deviation of the rule confidence in each rule set. The values are lower and closer to  $l - conf$  in our method, and this shows an advantage that the consequences of all rules can be accepted under uniform statistical belief levels.

**Table 1.** Performance of rule derivation

| $s - conf$ | Upper:Conventional method, Lower:Our method |          | ave. & std. of conf. |
|------------|---------------------------------------------|----------|----------------------|
|            | generated                                   | filtered |                      |
| 90.0%      | 3,695                                       | 578      | 2.90                 |
|            | 2,658                                       | 104      | 3.59                 |
| 70.0%      | 4,954                                       | 593      | 2.75                 |
|            | 2,046                                       | 141      | 2.86                 |
| 50.0%      | 7,470                                       | 455      | 2.76                 |
|            | 3,211                                       | 161      | 3.01                 |
| 30.0%      | 11,691                                      | 160      | 2.93                 |
|            | 2,915                                       | 136      | 3.08                 |

$l - sup$  is fixed to be 1.0%

## 4 Conclusion

A novel framework to mine association rules dedicated to estimation and prediction on the system underlying the data has been proposed in this paper. Its evaluation indicates 1) efficient reduction of redundant rules, 2) high ability and 3) uniform confidence for estimation and prediction. Our framework is expected to provide a new and effective tool for data mining in practical fields.

## References

- G. Piateksky-Shapiro.: Discovery, analysis, and presentation of strong rules. Knowledge Discovery in Databases. AAAI/MIT Press (1991)
- R.Agrwal and R.Srikant.: Fast algorithms for mining association rules. In *Proc. of the 20th VLDB Conference*, (1994) 487-499
- R.Kimball.: The Data Warehouse Toolkit. Wiley Computer Publishing (1996)
- U.Fayyad , G.Piateksky-Shapiro and P.Smyth.: From Data Mining to Knowledge Discovery in Databases. AI MAGAZINE FALL, (1996) 37-54
- R.Srikant, Q. Vu and R.Agrwal.: Mining Association Rules with Item Constraints. In *Proc. of 3rd Conference on Knowledge Discovery and Data Mining*, (1997) 67-73

# Rule Generalization by Condition Combination

Hanhong Xue and Qingsheng Cai

Department of Computer Science, University of Science and Technology of China  
Hefei, Anhui, People's Republic of China  
E-mail: s9511s04@sun10.cc.ustc.edu.cn, qscai@cs.ustc.edu.cn

**Abstract** This paper introduces an approach of condition combination to generate rules in a decision table. First we describe the concepts of negative condition and condition combination upon which our work is based, then their important properties for designing our algorithms. The algorithms are analyzed to show their time complexity and concern with attribute ordering.

## 1 Introduction

A branch of Knowledge Discovery in Databases is to induce classification rules from relational databases which are also viewed as information systems (Pawlak 1982).

**Definition 1** (Information System) An information system is a pair  $IS=(U, A)$  where  $U$  is a non-empty finite set called the universe,  $A$  is a non-empty finite set of attributes. Each attribute  $a \in A$  is a function  $a: U \rightarrow V_a$ , where  $V_a$  is the set of values of  $a$ , called the domain of  $a$ . Any non-empty subset  $B \subseteq A$  is a derived function  $B: U \rightarrow V_{b_1} \times \dots \times V_{b_k}$ . Given an object  $X \in U$  and a subset  $B = \{b_1, \dots, b_k\}$ , then  $B(X) = (a(b_1), \dots, a(b_k))$ .

**Definition 2** (Decision Table) A decision table is an information system  $DT=(U, C \cup \{d\})$  where  $d \notin C$ . Attributes in  $C$  are called condition attributes, and  $d$  the decision attribute. A condition is an assignment to the condition attributes.

**Definition 3** (Generalized Condition) When a condition does not depend on an attribute, the “don't care”, represented by “ $\varepsilon$ ”, is assigned to the attribute. A generalized condition is such an extended assignment to condition attributes.

**Definition 4** (Inferability) For two generalized conditions  $c_1$  and  $c_2$ ,  $c_2$  is inferable from  $c_1$ , denoted as  $c_1 \Rightarrow c_2$ , iff  $\forall a \in C, a(c_2) = a(c_1) \vee a(c_2) = \varepsilon$ .

**Definition 5** (Generalized Rule) A generalized rule consists of a generalized condition  $c_1$  and a decision  $q_1$ , and there is no object  $X \in U$  with condition  $c_2 = C(X)$  and decision  $q_2 = d(X)$  that  $c_2 \Rightarrow c_1$  and  $q_2 \neq q_1$ .

## 2 Condition Combination

**Definition 6** (Negative Condition) A negative condition is a generalized condition, which is impossible to appear in any generalized rules.

**Definition 7** (Maximal Negative Condition Set) is a set of negative conditions, in which no condition can be inferred from others and from which all negative conditions are inferable.

**Definition 8** (Condition Combination) We define an operator “ $\otimes$ ” for condition combination.  $c = c_1 \otimes c_2$  iff  $\forall a \in C, (a(c) = a(c_1) \wedge a(c) = a(c_2)) \vee (a(c) = \varepsilon \wedge a(c_1) \neq a(c_2))$ . A “don't-care” is unequal to all other values.

**Theorem 1** Given generalized conditions  $c$ ,  $c_1$  and  $c_2$ , if  $c_1 \Rightarrow c \wedge c_2 \Rightarrow c$ , then  $c_1 \otimes c_2 \Rightarrow c$ .

## 3 Algorithms and Analyses

**Theorem 2** For any negative condition, there exists a pair of objects with different decisions that the combination of their conditions infers the negative condition.

**Algorithm 1** Create  $MNCS$ 

```

1 $MNCS \leftarrow \emptyset$
2 for $X, Y \in U, d(X) \neq d(Y)$ do
3 $c \leftarrow C(X) \otimes C(Y)$
4 if $\sim(\exists c' \in MNCS, c' \Rightarrow c)$ then
5 $IR \leftarrow \{c' \in MNCS \mid c' \Rightarrow c\}$
6 $MNCS \leftarrow MNCS \cup \{c\} \setminus IR$
7 end if
8 end for

```

According to Theorem 2, we design Algorithm 1 to create  $MNCS$  for a decision table. The worst time complexity of Algorithm 1 is  $O(mu^4)$ , where  $u=|U|$  and  $m=|C|$ . The candidates of generalized rules ( $CGR$ ) can also be generated by combining conditions of objects with the same decision.

**Algorithm 2** Generate  $CGR$ 

```

1 $CGR \leftarrow \{X \in U \mid d(X)=q\}$
2 repeat
3 $RR \leftarrow CGR$
4 $changed \leftarrow \text{false}$
5 for $X, Y \in RR, X \neq Y$ do
6 $c \leftarrow C(X) \otimes C(Y)$
7 if $\sim(\exists c' \in MNCS, c' \Rightarrow c) \wedge \sim(\exists c' \in CGR, c \Rightarrow c')$ then
8 $IR \leftarrow \{c' \in CGR \mid c' \Rightarrow c\}$
9 if $IR \neq \emptyset$ then
10 $CGR \leftarrow CGR \cup \{c\} \setminus IR$
11 $changed \leftarrow \text{true}$
12 end if
13 end if
14 end for
15 until $\text{not } changed$

```

**Theorem 3** In algorithm 2, a generalized condition  $c$  with  $k$  “don’t-care”s will be generated no later than the  $\lceil 1 + \log k \rceil$ -th iteration. Let  $m=|C|$ ,  $n=|\{X \in U \mid d(X)=q\}|$ ,  $g=|MNCS|$ . The worst time complexity of algorithm 2 is  $O(m(n+g)n^3 \log m)$ .

**Theorem 4** The number of rules in  $CGR$  cannot be further reduced by dropping attributes of those rules.

While other algorithms dealing directly with decision tables are significantly influenced by the order in which condition attributes are arranged (Ziarko 1995), such influence on our algorithm is little. Some more effort will be needed to obtain maximally generalized rules (Cercone 1997).

## References

- Pawlak, Z. 1982. Rough Sets, *Information and Computer Science*, 11(5), 341-356
- Ziarko, W. and Shan, N. 1995. Knowledge discovery as a search for classification, Workshop on Rough Sets and Database Mining, 23rd Annual Computer Science, CSC'95
- Cercone, N., Hamilton, H. J., Hu, X. and Shan, N. 1997. Data mining using attribute-oriented generalization and information reduction, in Rough sets and data mining: analysis for imprecise data, T.Y.Lin and N.Cercone eds., Kluwer Academic Publishers.

# Author Index

- Akimori, H. 399  
Albrecht, D. W. 1  
Allison, L. 380  
Anand , S. S. 13, 25  
Baxter, R. A. 87, 222  
Bell, D.A. 25  
Brenot, P. 385  
Büchter, O. 36  
Caelli, T. 407  
Cai, Q. 420  
Carter, C. L. 159  
Carver, D. L. 405  
Cercone, N. 159  
Chang, C.-H. 374  
Chen, H. 394  
Cheung, D. W. 48  
Cho, V. 376  
Cole, R. 378  
Coomans, D. 322  
Crémilleux, B. 258  
Dai, H. 61  
Di, K. 392  
Dong, G. 72  
Dong, J. 360  
Dowe, D. L. 87, 96, 380  
Edwards, R. T. 96  
Eklund, P. 378  
Estivill-Castro, V. 110  
Fraser, S. F. 409  
Frayman, Y. 122  
Fulcher, J. 390  
Gray, B. 132  
Hamilton, H. J. 159  
Hamish, S. M. 409  
Han, J. 144  
Hegland, M. 401  
Hilderman, R. J. 159  
Ho , K. M. 383  
Hsieh, W.-S. 271  
Hsu, C.-C. 374  
Huang, Z. 401  
Hughes, J. G. 13, 25  
Iizuka, T. 174  
Iizuka, Y. 174  
Isobe, S. 174  
Jacquenet, F. 385  
Jagielska, I. 388  
Kindermann, J. 234  
Kitsuregawa, M. 283  
Kohane, I. S. 409  
Koperski, K. 144  
Koshizen, T. 390  
Kowalczyk, W. 186  
Kryszkiewicz, M. 198  
Li, Daren 392  
Li, Deyi 392  
Li, J. 72  
Liu, H. 210, 397  
Liu, Y. 394  
Lu, H. 210  
Lu, Y. 397  
Maeda, A. 399  
Maki, H. 399  
Matsuura, H. 417  
Monks de Oca, C. 405  
Motoda, H. 417  
Murray, A. T. 110  
Ng, M.K. 401  
Nicholson, A. E. 1

- Nowak, C. 403  
 Ogawa, H. 390  
 Ohbo, N. 394  
 Ohsuga, S. 360  
 Oliver, J. J. 87, 222  
 Orlowska, M. E. 132  
 Paass, G. 234  
 Patrick, S. 322  
 Patterson, D. 25  
 Pearce, A. R. 407  
 Perrin, P. 246  
 Petry, F. 246  
 Piasta, Z. 186  
 Pringle, G. 380  
 Ragel, A. 258  
 Scott, P. D. 383  
 Shao, S.-C. 271  
 Sher, B.-Y. 271  
 Shi, X. 392  
 Shintani, T. 283  
 Shiohara, H. 174  
 Sever, H. 310  
 Siebes, A. 295  
 Stefanovic, N. 144  
 Stranieri, A. 336  
 Struzik, Z. R. 295  
 Tan, C. L. 397  
 Timmermans, H. 412  
 Tolun, M. R. 310  
 Tsien, C. L. 409  
 Uludağ, M. 310  
 Vanthienen, J. 412  
 Vazirgiannis, M. 414  
 de Vel, O. 322  
 Walker, D. 378  
 Wallace, C. S. 87, 222  
 Wang, L. 122  
 Washio, T. 417  
 Wets, G. 412  
 Wirth, R. 35  
 Wüthrich, B. 376  
 Xiao, Y. 48  
 Xue, H. 420  
 Yao, J. 210  
 Yu, J. X. 394  
 Zeleznikow, J. 336  
 Zheng, Z. 348  
 Zhong, N. 360  
 Zukerman, I. 1

# Lecture Notes in Artificial Intelligence (LNAI)

- Vol. 1224: M. van Someren, G. Widmer (Eds.), Machine Learning: ECML-97. Proceedings, 1997. XI, 361 pages. 1997.
- Vol. 1227: D. Galmiche (Ed.), Automated Reasoning with Analytic Tableaux and Related Methods. Proceedings, 1997. XI, 373 pages. 1997.
- Vol. 1228: S.-H. Nienhuys-Cheng, R. de Wolf, Foundations of Inductive Logic Programming. XVII, 404 pages. 1997.
- Vol. 1229: G. Kraetzschmar, Distributed Reason Maintenance for Multiagent Systems. XIV, 296 pages. 1997.
- Vol. 1236: E. Maier, M. Mast, S. LuperFoy (Eds.), Dialogue Processing in Spoken Language Systems. Proceedings, 1996. VIII, 220 pages. 1997.
- Vol. 1237: M. Boman, W. Van de Velde (Eds.), Multi-Agent Rationality. Proceedings, 1997. XII, 254 pages. 1997.
- Vol. 1244: D. M. Gabbay, R. Kruse, A. Nonnengart, H.J. Ohlbach (Eds.), Qualitative and Quantitative Practical Reasoning. Proceedings, 1997. X, 621 pages. 1997.
- Vol. 1249: W. McCune (Ed.), Automated Deduction – CADE-14. Proceedings, 1997. XIV, 462 pages. 1997.
- Vol. 1257: D. Lukose, H. Delugach, M. Keeler, L. Searle, J. Sowa (Eds.), Conceptual Structures: Fulfilling Peirce's Dream. Proceedings, 1997. XII, 621 pages. 1997.
- Vol. 1263: J. Komorowski, J. Zytkow (Eds.), Principles of Data Mining and Knowledge Discovery. Proceedings, 1997. IX, 397 pages. 1997.
- Vol. 1266: D.B. Leake, E. Plaza (Eds.), Case-Based Reasoning Research and Development. Proceedings, 1997. XIII, 648 pages. 1997.
- Vol. 1265: J. Dix, U. Furbach, A. Nerode (Eds.), Logic Programming and Nonmonotonic Reasoning. Proceedings, 1997. X, 453 pages. 1997.
- Vol. 1285: X. Jao, J.-H. Kim, T. Furuhashi (Eds.), Simulated Evolution and Learning. Proceedings, 1996. VIII, 231 pages. 1997.
- Vol. 1286: C. Zhang, D. Lukose (Eds.), Multi-Agent Systems. Proceedings, 1996. VII, 195 pages. 1997.
- Vol. 1297: N. Lavrač, S. Džeroski (Eds.), Inductive Logic Programming. Proceedings, 1997. VIII, 309 pages. 1997.
- Vol. 1299: M.T. Pazienza (Ed.), Information Extraction. Proceedings, 1997. IX, 213 pages. 1997.
- Vol. 1303: G. Brewka, C. Habel, B. Nebel (Eds.), KI-97: Advances in Artificial Intelligence. Proceedings, 1997. XI, 413 pages. 1997.
- Vol. 1307: R. Kompe, Prosody in Speech Understanding Systems. XIX, 357 pages. 1997.
- Vol. 1314: S. Muggleton (Ed.), Inductive Logic Programming. Proceedings, 1996. VIII, 397 pages. 1997.
- Vol. 1316: M. Li, A. Maruoka (Eds.), Algorithmic Learning Theory. Proceedings, 1997. XI, 461 pages. 1997.
- Vol. 1317: M. Leman (Ed.), Music, Gestalt, and Computing. IX, 524 pages. 1997.
- Vol. 1319: E. Plaza, R. Benjamins (Eds.), Knowledge Acquisition, Modelling and Management. Proceedings, 1997. XI, 389 pages. 1997.
- Vol. 1321: M. Lenzerini (Ed.), AI\*IA 97: Advances in Artificial Intelligence. Proceedings, 1997. XII, 459 pages. 1997.
- Vol. 1323: E. Costa, A. Cardoso (Eds.), Progress in Artificial Intelligence. Proceedings, 1997. XIV, 393 pages. 1997.
- Vol. 1325: Z.W. Raś, A. Skowron (Eds.), Foundations of Intelligent Systems. Proceedings, 1997. XI, 630 pages. 1997
- Vol. 1328: C. Retoré (Ed.), Logical Aspects of Computational Linguistics. Proceedings, 1996. VIII, 435 pages. 1997.
- Vol. 1342: A. Sattar (Ed.), Advanced Topics in Artificial Intelligence. Proceedings, 1997. XVIII, 516 pages. 1997.
- Vol. 1348: S. Steel, R. Alami (Eds.), Recent Advances in AI Planning. Proceedings, 1997. IX, 454 pages. 1997.
- Vol. 1359: G. Antinou, A. Ghose, M. Truszczyński (Eds.), Learning and Reasoning with Complex Representations. Proceedings, 1996. X, 283 pages. 1998.
- Vol. 1360: D. Wang (Ed.), Automated Deduction in Geometry. Proceedings, 1996. VII, 235 pages. 1998.
- Vol. 1365: M.P. Singh, A. Rao, M.J. Wooldridge (Eds.), Intelligent Agents IV. Proceedings, 1997. XII, 351 pages. 1998.
- Vol. 1371: I. Wachsmuth, M. Fröhlich (Eds.), Gesture and Sign-Language in Human-Computer Interaction. Proceedings, 1997. XI, 309 pages. 1998.
- Vol. 1374: H. Bunt, R.-J. Beun, T. Borghuis (Eds.), Multimodal Human-Computer Communication. VIII, 345 pages. 1998.
- Vol. 1387: C. Lee Giles, M. Gori (Eds.), Adaptive Processing of Sequences and Data Structures. Proceedings, 1997. XII, 434 pages. 1998.
- Vol. 1394: X. Wu, R. Kotagiri, K.B. Korb (Eds.), Research and Development in Knowledge Discovery and Data Mining. Proceedings, 1998. XVI, 424 pages. 1998.
- Vol. 1397: H. de Swart (Ed.), Automated Reasoning with Analytic Tableaux and Related Methods. Proceedings, 1998. X, 325 pages. 1998.
- Vol. 1398: C. Nédellec, C. Rouveiro (Eds.), Machine Learning: ECML-98. Proceedings, 1998. XII, 420 pages. 1998.

# Lecture Notes in Computer Science

- Vol. Vol. 1359: G. Antinou, A. Ghose, M. Truszcynski (Eds.), Learning and Reasoning with Complex Representations. Proceedings, 1996. X, 283 pages. 1998. (Subseries LNAI).
- 1360: D. Wang (Ed.), Automated Deduction in Geometry. Proceedings, 1996. VII, 235 pages. 1998. (Subseries LNAI).
- Vol. 1361: B. Christianson, B. Crispo, M. Lomas, M. Roe (Eds.), Security Protocols. Proceedings, 1997. VIII, 217 pages. 1998.
- Vol. 1362: D.K. Panda, C.B. Stunkel (Eds.), Network-Based Parallel Computing. Proceedings, 1998. X, 247 pages. 1998.
- Vol. 1363: J.-K. Hao, E. Lutton, E. Ronald, M. Schoenauer, D. Snyers (Eds.), Artificial Evolution. XI, 349 pages. 1998.
- Vol. 1364: W. Conen, G. Neumann (Eds.), Coordination Technology for Collaborative Applications. VIII, 282 pages. 1998.
- Vol. 1365: M.P. Singh, A. Rao, M.J. Wooldridge (Eds.), Intelligent Agents IV. Proceedings, 1997. XII, 351 pages. 1998. (Subseries LNAI).
- Vol. 1367: E.W. Mayr, H.J. Prömel, A. Steger (Eds.), Lectures on Proof Verification and Approximation Algorithms. XII, 344 pages. 1998.
- Vol. 1368: Y. Masunaga, T. Katayama, M. Tsukamoto (Eds.), Worldwide Computing and Its Applications — WWCA'98. Proceedings, 1998. XIV, 473 pages. 1998.
- Vol. 1370: N.A. Streitz, S. Konomi, H.-J. Burkhardt (Eds.), Cooperative Buildings. Proceedings, 1998. XI, 267 pages. 1998.
- Vol. 1371: I. Wachsmuth, M. Fröhlich (Eds.), Gesture and Sign-Language in Human-Computer Interaction. Proceedings, 1997. XI, 309 pages. 1998. (Subseries LNAI).
- Vol. 1372: S. Vaudenay (Ed.), Fast Software Encryption. Proceedings, 1998. VIII, 297 pages. 1998.
- Vol. 1373: M. Morvan, C. Meinel, D. Krob (Eds.), STACS 98. Proceedings, 1998. XV, 630 pages. 1998.
- Vol. 1374: H. Bunt, R.-J. Beun, T. Borghuis (Eds.), Multimodal Human-Computer Communication. VIII, 345 pages. 1998. (Subseries LNAI).
- Vol. 1375: R. D. Hersch, J. André, H. Brown (Eds.), Electronic Publishing, Artistic Imaging, and Digital Typography. Proceedings, 1998. XIII, 575 pages. 1998.
- Vol. 1376: F. Parisi Presicce (Ed.), Recent Trends in Algebraic Development Techniques. Proceedings, 1997. VIII, 435 pages. 1998.
- Vol. 1377: H.-J. Schek, F. Saltor, I. Ramos, G. Alonso (Eds.), Advances in Database Technology — EDBT'98. Proceedings, 1998. XII, 515 pages. 1998.
- Vol. 1378: M. Nivat (Ed.), Foundations of Software Science and Computation Structures. Proceedings, 1998. X, 289 pages. 1998.
- Vol. 1379: T. Nipkow (Ed.), Rewriting Techniques and Applications. Proceedings, 1998. X, 343 pages. 1998.
- Vol. 1380: C.L. Lucchesi, A.V. Moura (Eds.), LATIN'98: Theoretical Informatics. Proceedings, 1998. XI, 391 pages. 1998.
- Vol. 1381: C. Hankin (Ed.), Programming Languages and Systems. Proceedings, 1998. X, 283 pages. 1998.
- Vol. 1382: E. Astesiano (Ed.), Fundamental Approaches to Software Engineering. Proceedings, 1998. XII, 331 pages. 1998.
- Vol. 1383: K. Koskimies (Ed.), Compiler Construction. Proceedings, 1998. X, 309 pages. 1998.
- Vol. 1384: B. Steffen (Ed.), Tools and Algorithms for the Construction and Analysis of Systems. Proceedings, 1998. XIII, 457 pages. 1998.
- Vol. 1385: T. Margaria, B. Steffen, R. Rückert, J. Posegga (Eds.), Services and Visualization. Proceedings, 1997/1998. XII, 323 pages. 1998.
- Vol. 1386: T.A. Henzinger, S. Sastry (Eds.), Hybrid Systems: Computation and Control. Proceedings, 1998. VIII, 417 pages. 1998.
- Vol. 1387: C. Lee Giles, M. Gori (Eds.), Adaptive Processing of Sequences and Data Structures. Proceedings, 1997. XII, 434 pages. 1998. (Subseries LNAI).
- Vol. 1388: J. Rolim (Ed.), Parallel and Distributed Processing. Proceedings, 1998. XVII, 1168 pages. 1998.
- Vol. 1389: K. Tombre, A.K. Chhabra (Eds.), Graphics Recognition. Proceedings, 1997. XII, 421 pages. 1998.
- Vol. 1391: W. Banzhaf, R. Poli, M. Schoenauer, T.C. Fogarty (Eds.), Genetic Programming. Proceedings, 1998. X, 232 pages. 1998.
- Vol. 1393: D. Bert (Ed.), B'98: Recent Advances in the Development and Use of the B Method. Proceedings, 1998. VIII, 313 pages. 1998.
- Vol. 1394: X. Wu, R. Kotagiri, K.B. Korb (Eds.), Research and Development in Knowledge Discovery and Data Mining. Proceedings, 1998. XVI, 424 pages. 1998. (Subseries LNAI).
- Vol. 1396: G. Davida, M. Mambo, E. Okamoto (Eds.), Information Security. Proceedings, 1997. XII, 357 pages. 1998.
- Vol. 1397: H. de Swart (Ed.), Automated Reasoning with Analytic Tableaux and Related Methods. Proceedings, 1998. X, 325 pages. 1998. (Subseries LNAI).
- Vol. 1371: I. Wachsmuth, M. Fröhlich (Eds.), Gesture and Sign-Language in Human-Computer Interaction. Proceedings, 1997. XI, 309 pages. 1998. (Subseries LNAI).